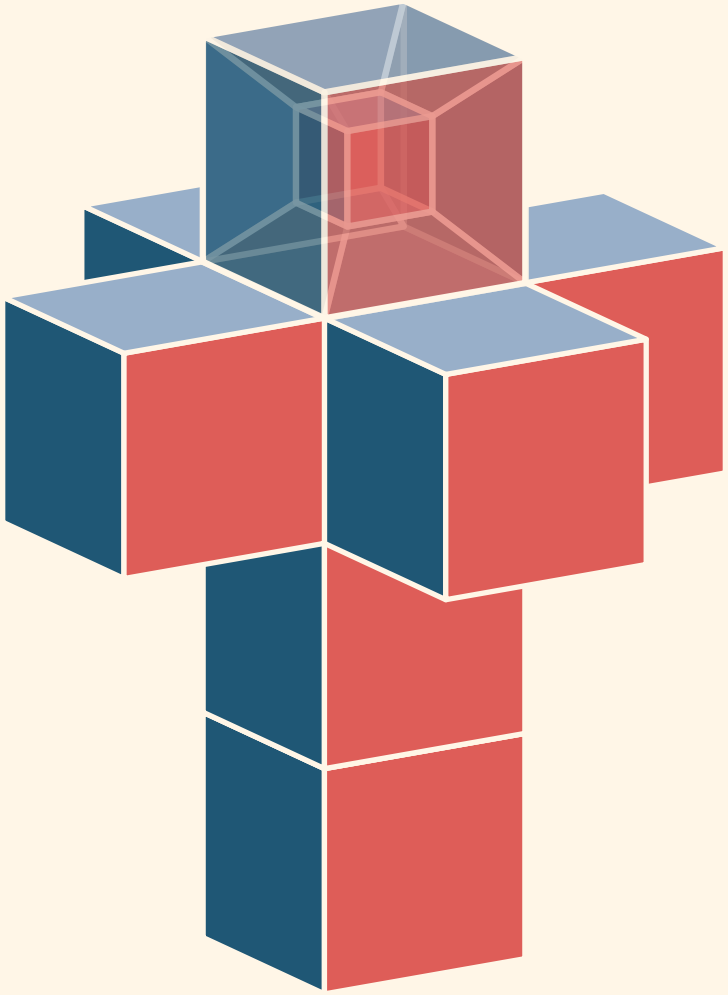
Unfolding the Infinite

Het ontwikkelen van n-dimensionale polytopen naar (n-1)-dimensionale netstructuren



Bram Leisink & Johnathan Yazvin

Mondial College

Profielwerkstuk

Wiskunde

Unfolding the Infinite

Het ontwikkelen van n-dimensionale polytopen naar $(n-1)$-dimensionale netstructuren

---

Auteurs
Bram Leisink
Johnathan Yazvin

Begeleider
Joris Coenders

6 december 2024

Copyright en Dankwoord

Lisa Simpson:
"Well, where's my dad?"

Prof. John Frink:
"Well, it should be clear to even the most dimwitted individual—who holds an advanced degree in hypothetical topology—that Homer Simpson has stumbled into...the Third Dimension!"

(The Simpsons, S7 E6: "Treehouse of Horror VI", 1995)

Met speciale dank aan Noah Reijnen – onze eindredacteur, fotograaf en spellingscontrole – voor zijn nauwkeurigheid, kritische blik en waardevolle hulp bij het afronden van dit project.

Voorwoord

Beste lezer,

Wiskunde is veel meer dan een droge verzameling formules, theorieën en bewijzen. Het is een reis vol ontdekkingen, verrassingen en soms zelfs frustraties. Die reis wordt een stuk spannender als je ervoor kiest om je profielwerkstuk te schrijven over een onderwerp waar je aan het begin precies niets vanaf weet.

Wij besloten ons avontuur te wijden aan de mysterieuze wereld van n-dimensionale polytopen en hun ontwikkeling naar $(n-1)$-dimensionale netstructuren. Klinkt ingewikkeld? Dat is het ook. Maar juist die complexiteit maakte het een uitdaging die we niet konden weerstaan. Want hoe leg je een vierdimensionaal object uit op een stuk tweedimensionaal papier, terwijl je leeft in een driedimensionaal universum? Het antwoord: met veel geduld, nieuwsgierigheid en een flinke portie doorzettingsvermogen.

Dit werkstuk is echter niet alleen ons werk. We willen van deze gelegenheid gebruik maken om iedereen te bedanken die op welke manier dan ook heeft bijgedragen. Onze begeleider, Joris Coenders, die ons met enkele waardevolle inzichten op weg heeft geholpen, ondanks dat zelfs hij af en toe niet meer precies begreep waar we mee bezig waren. Onze vrienden en familie, die (soms tegen hun wil) geduldig luisterden terwijl we probeerden uit te leggen wat een 4-dimensionale polytoop eigenlijk is. In het speciaal Noah Reijnen, die het voor elkaar kreeg om in elke versie toch nog fouten te vinden. En natuurlijk u, de lezer, die de moed heeft om dit werk in handen te nemen.

We hopen dat u met net zoveel plezier zult lezen als waarmee wij dit profielwerkstuk hebben geschreven.

Bram Leisink & Johnathan Yazvin
Nijmegen, 6 december 2024

Samenvatting

In dit profielwerkstuk presenteren wij een systematische methode voor het ontwikkelen van n-dimensionale polytopen naar $(n-1)$-dimensionale netstructuren. Deze aanpak biedt niet alleen een theoretisch kader, maar ook een praktisch toepasbaar algoritme dat succesvol is getest.

Onze methode, die gebaseerd is op fundamentele eigenschappen van polytopen, combineert inzichten uit grafentheorie en computationele geometrie. Het resultaat is een robuust algoritme dat in principe op elke polytoop toepasbaar is, ongeacht zijn vorm of complexiteit. De n-kubus diende als waardevolle casestudy, waarbij we hebben aangetoond hoe hogere-dimensionale structuren met ons algoritme kunnen worden 'uitgevouwen' naar lagere dimensies.

Een belangrijk doel van ons onderzoek was om hogere-dimensionale wiskunde toegankelijker te maken voor een breed publiek. Door abstracte concepten te visualiseren via netstructuren hebben we aangetoond dat complexe wiskundige objecten intuïtief begrepen kunnen worden. De positieve reacties op onze presentaties en workshops bevestigen de educatieve waarde van onze benadering. Ter ondersteuning hebben we een website (`pws.bramleisink.nl`) ontwikkeld met aanvullende visualisaties en onderwijsmateriaal.

De toepassingen van dit onderzoek strekken zich uit over diverse gebieden. In computergraphics kan onze methode helpen bij het visualiseren van hogere-dimensionale data, essentieel voor machine learning en kunstmatige intelligentie. In de natuurkunde biedt het toepassingen in kristallografie, moleculaire modellering en snaartheorie. Daarnaast zijn er praktische toepassingsmogelijkheden in verpakkingsontwerp en biomedische wetenschap.

Ondanks onze successen erkennen we de beperkingen van onze aanpak. Het visualiseren van objecten uit hogere dimensies blijft uitdagend, en niet elke polytoop heeft noodzakelijkerwijs een overlapvrij net. Deze uitdagingen, samen met enkele open vragen, bieden interessante richtingen voor toekomstig onderzoek.

Dit werk vormt een solide fundament op het kruispunt van multidimensionale meetkunde, grafentheorie en computerwetenschap, en opent nieuwe wegen voor zowel theoretisch als praktisch onderzoek naar de fascinerende wereld van hogere-dimensionale geometrie.

Wiskundige notatie en definities

Dit hoofdstuk bevat de belangrijkste notatie en een referentie naar definities voor dit profielwerkstuk. Een uitgebreide woordenlijst is aan het einde van dit werk te vinden.

Notatie

n Dimensie van de polytoop.

P_n Een n-dimensionale polytoop.

$\mathbb{R}^n$ De n-dimensionale euclidische ruimte.

$\boldsymbol{v}$ Een vector.

$\boldsymbol{u} \cdot \boldsymbol{v}$ Het inproduct van twee vectoren.

$(x_1, x_2, \ldots)$ Een geordende tuple.

$\{x_1, x_2, \ldots\}$ Een ongeordende verzameling.

V De verzameling hoekpunten (in een polytoop).

V_i Het hoekpunt met index i.

$\#V$ Het aantal hoekpunten.

E De verzameling lijnstukken (in een polytoop).

E_i Het lijnstuk met index i.

$\#E$ Het aantal lijnstukken.

F_k De verzameling k-dimensionale facetten (in een polytoop).

$F_{k,i}$ Het k-dimensionale facet met index i.

$\#F_k$ Het aantal k-dimensionale facetten.

G Een graaf.

V De verzameling knopen (in een graaf).

E De verzameling randen (in een graaf).

(u, v) De rand tussen knopen u en v.

$u \sim v$ De knopen u en v zijn aanliggend in de graaf.

$w(u, v)$ Het gewicht van de rand (u, v).

$\deg(v)$ De graad van knoop v.

$\deg^-(v)$ De in-graad van knoop v.

$\deg^+(v)$ De uit-graad van knoop v.

$O(f(n))$ De tijdcomplexiteit in termen van functie $f(n)$.

$\binom{n}{k}$ De binomiale coëfficiënt, het aantal manieren om k objecten te kiezen uit n zonder teruglegging.

χ De Euler-karakteristiek.

$\mathrm{Aff}(V_1, V_2, \ldots)$ De affiene deelruimte van een verzameling vertexen.

Definities

In dit werk worden de gebruikte wiskundige definities en concepten, zoals de definitie van een polytoop, kubus en andere gerelateerde termen, gebaseerd op enkele standaardwerken in de wiskunde. Deze bronnen, die als referentie dienen voor de basisbegrippen, worden hier genoemd, maar zullen verder in het werk niet telkens expliciet worden geciteerd. De belangrijkste bronnen voor deze definities zijn:

- Coxeter, H. S. M., *Regular Polytopes* (1973) [11]
- Grinberg, M., *Polytopes, Graphs, and Complexes* (2006) [21]
- Bondy, J. A., & Murty, U. S. R., *Graph Theory* (2008) [6]
- Coxeter, H. S. M., *Introduction to Geometry* (1989) [9]

Inhoudsopgave

Wiskundige notatie en definities **7**
Notatie 7
Definities 8

1 Inleiding **13**
1.1 Probleemstelling 13
1.2 Doelstelling 14
1.3 Opbouw 15

I Theoretisch kader **17**

2 Euclidische ruimtes **19**
2.1 Vertexen en coördinaten 19
2.1.1 Getallenlijn 20
2.1.2 Coördinatenvlak 20
2.1.3 Driedimensionaal coördinatenstelsel 22
2.1.4 Hogerdimensionaal coördinatenstelsel 23
2.2 Meetkundige eigenschappen 26
2.2.1 Afstand tussen punten 26
2.2.2 Inproduct 26
2.3 Projecties 27
2.3.1 Orthogonale projectie 28
2.3.2 Perspectiefprojectie 28
2.3.3 Complexe projecties 29

3 Polytopen **31**
3.1 Definitie 31
3.1.1 Facetten van een polytoop 32
3.2 Elementen van een polytoop 32
3.3 4D en hoger 33
3.4 Eigenschappen van polytopen 34

3.4.1 De klassieke formule: 2D en 3D . . . 34
3.4.2 Uitbreiding naar hogere dimensies . . . 35
3.4.3 Waarom is dit belangrijk? . . . 36
3.5 Polytopen in een affiene deelruimte . . . 36
3.5.1 Berekenen van de dimensionaliteit van een polytoop . . . 37

4 Netstructuren **39**
4.1 Definitie en eigenschappen . . . 39
4.1.1 Behoud van eigenschappen . . . 39
4.2 Netstructuren in hogere dimensies . . . 41
4.3 Uitvouwregels en eigenschappen van netstructuren . . . 41
4.4 Overlap in netstructuren . . . 42
4.4.1 Niet-overlappende netstructuren . . . 42
4.4.2 Overlappende netstructuren . . . 43
4.4.3 Existentie van niet-overlappende netstructuren . . . 43

5 Grafentheorie **45**
5.1 Inleiding . . . 45
5.1.1 Wat is grafentheorie? . . . 45
5.1.2 Geschiedenis van grafentheorie . . . 45
5.2 Basisconcepten . . . 46
5.2.1 Grafen . . . 46
5.2.2 Gerichte en ongerichte grafen . . . 46
5.2.3 Gewogen en ongewogen grafen . . . 46
5.2.4 Knopen en randen . . . 47
5.2.5 Graad van een knoop . . . 47
5.3 Grafen representaties . . . 47
5.3.1 Adjacency list . . . 48
5.3.2 Adjacency matrix . . . 49
5.3.3 Incidence matrix . . . 49
5.4 Spanning trees . . . 50
5.5 Minimal spanning trees . . . 51
5.5.1 Wat is een minimal spanning tree? . . . 51
5.5.2 Algoritmen voor minimal spanning trees . . . 52
5.5.3 Toepassingen van minimal spanning trees . . . 52

6 Casestudy deel 1: De n-kubus **53**
6.1 Recursieve definitie van de n-kubus . . . 53
6.1.1 Voorbeelden van de definitie . . . 53
6.1.2 Concrete definitie . . . 56

6.2 n-kubus als oplossing van lineaire ongelijkheden 57
6.2.1 Voorbeeld: 2-kubus (vierkant) 57
6.2.2 Algemeen geval: n-kubus 58
6.3 De elementen van de n-kubus 58
6.4 De coördinaten van de hoekpunten in een n-kubus 60

II De methode 63

7 Inleiding tot de methode 65
7.1 Is er een net voor elke polytoop? 65
7.1.1 Netten zonder overlap 65
7.1.2 Netten met overlap 65
7.2 Opbouw van de methode 66

8 Representatie polytoop als een graaf 67
8.1 Recursief: vooruit 68
8.2 Recursief: achteruit 69
8.3 Wanneer grenzen facetten aan elkaar? 71
8.4 Efficiëntere representatie van een polytoop 72

9 Vinden van de optimale spanning tree 73
9.1 Waarom de methode resulteert in een netstructuur 73
9.2 Van graaf naar spanning tree 74
9.3 Tree met minimale overlap 74
9.3.1 Gewichten toekennen 74

10 Van spanning tree naar netstructuur 75
10.1 Facet verplaatsen naar lagere dimensie 75
10.1.1 Controleren van de dimensie van het facet 75
10.1.2 Zwaartepunt van facet op oorsprong 76
10.1.3 Roteren naar een standaardvorm 76
10.1.4 Projectie naar lagere dimensie 78
10.2 Facet verplaatsen naar andere facet 78

11 Casestudy deel 2: Het ontwikkelen van de n-kubus 79
11.1 De 3-kubus 79
11.1.1 Vinden van de spanning tree 85
11.2 De 7-kubus 89

III Afsluiting **91**

12 Discussie &Conclusie **93**
12.1 De educatieve waarde van ons onderzoek 93
12.2 Beperkingen van de theorie en methoden 94
12.3 De casestudy van de n-kubus 94
12.4 Toepassingen in de praktijk 95
12.5 Toekomstig onderzoek 95
12.6 Conclusie 96

Woordenlijst **97**

Bibliografie **103**

A Afstand tussen punten en het inproduct **107**

B Dimensionaliteit van een polytoop berekenen **109**
B.1 Algebraïsch 109
B.2 In Python 112

C Suggesties voor verder onderzoek **113**
C.1 Is elke polytoop te definiëren door slechts hoekpunten en hun relaties? 113
C.2 Is er een functie $f(n)$ voor het aantal unieke netstructuren van een n-kubus? 115
C.3 Is elke convexe n-dimensionale polytoop te ontwikkelen naar een netstructuur zonder overlap? 116
C.4 Zijn er naast de tetraëder, andere polytopen die een netstructuur kunnen hebben met een convexe buitenste rand? 117
C.5 Wat zijn heuristieken om overlap in hoger-dimensionale netstructuren te voorkomen? 117

D Gerelateerde materialen **119**

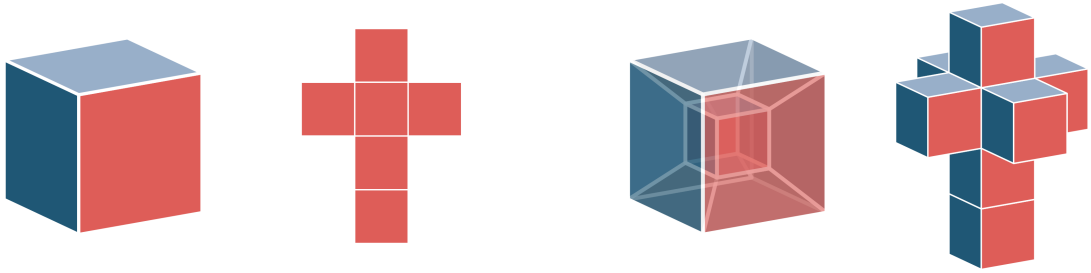
Hoofdstuk 1

Inleiding

1.1 Probleemstelling

Het concept van hogere dimensies is een van de fundamentele concepten binnen de wiskunde en natuurkunde. In het dagelijks leven hebben we te maken met drie dimensies: lengte, breedte en hoogte. Echter, in de theoretische wis- en natuurkunde komen vaak hogere dimensies voor, zoals in het geval van de vierdimensionale ruimtetijd in de relativiteitstheorie [40] of in de hoge-dimensionale ruimtes waarin datasets voor kunstmatige intelligentie zich bevinden [35]. Hoewel deze concepten abstract kunnen zijn, bieden ze krachtige modellen om te werken met complexe systemen.

Dit profielwerkstuk richt zich op de *multidimensionale meetkunde* en een specifieke uitdaging binnen dit vakgebied: het ontwikkelen van n-dimensionale polytopen naar $(n-1)$-dimensionale netstructuren. Een polytoop is een object dat in elke dimensie kan bestaan. Polytopen kom je overal tegen, zoals in de derde dimensie in de vorm van kubussen of piramides, of in de tweede dimensie als zeshoek of ruit. Een netstructuur (ook wel uitslag genoemd) is een representatie van deze polytoop, nadat deze ontwikkeld is naar een lagere dimensie, zoals in fig. 1.1.



(a) Een 2D netstructuur van de 3D kubus

(b) Een 3D netstructuur van de 4D kubus

Figuur 1.1: Het ontwikkelen van een polytoop

Door zo'n n-dimensionale polytoop te ontwikkelen, kunnen we inzicht krijgen in

de onderliggende patronen en structuren, en deze visualiseren op een lagere dimensie. Dit proces wordt vaak toegepast op 3D-objecten die worden ontwikkeld naar 2D-structuren (zie fig. 1.1a), bijvoorbeeld bij het maken van vouwbare modellen [39] voor architectuur of andere creatieve doeleinden. De studie naar het ontwikkelen van polyhedra (driedimensionale polytopen) werd gepopulariseerd door Albrecht Dürer in de vroege 16e eeuw, met zijn boek *The Painter's Manual*, waar hij de eerste voorbeelden van netstructuren presenteerde.

Toch wordt er nog weinig onderzoek gedaan naar het ontwikkelen van polytopen in hogere dimensies (zoals 4D) naar hun respectievelijk lagere dimensies. In de gevallen waar dit wel gebeurt, wordt er meestal slechts gekeken naar een specifieke situatie of dimensie. Zo wordt er vaak alleen gekeken naar convexe polytopen [38], de vierde dimensie [20][36] of objecten met speciale eigenschappen (zoals de n-kubus) [17].

1.2 Doelstelling

Waarom zou iemand zich bezighouden met het ontwikkelen van een n-dimensionale polytoop? Op het eerste gezicht lijkt dit een puur theoretisch probleem, iets voor wiskundigen die ver verwijderd zijn van de praktijk. Toch heeft het ontwikkelen van objecten naar een lagere dimensie verrassend veel toepassingen in de echte wereld.

Denk aan verpakkingsontwerp: de manier waarop een kartonnen doos plat is voordat deze in elkaar wordt gezet, wordt verkregen door een 3D-naar-2D ontwikkeling. Dit principe wordt ook gebruikt in de productie-industrie, waar metalen platen worden gesneden en vervolgens tot complexe vormen worden gevouwen. Maar wat als we verder gaan dan drie dimensies? In de biomedische wetenschap wordt bijvoorbeeld gekeken naar het vouwen van complexe moleculen en eiwitten, waarbij een 4D-benadering kan helpen om ziekten beter te begrijpen en effectievere behandelingen te ontwikkelen. Ook in datawetenschap speelt dit concept een rol: algoritmes moeten vaak complexe, hoge-dimensionale data op een begrijpelijke manier presenteren—iets wat in essentie lijkt op het ontwikkelen van een polytoop.

Naast de praktische toepassingen heeft dit probleem een sterke educatieve waarde. Het biedt een visuele en intuïtieve manier om abstracte concepten uit de wiskunde te begrijpen. De overgang van 3D naar 2D is voor veel mensen goed voorstelbaar, en juist die concrete visualisatie maakt het mogelijk om door te denken naar hogere dimensies.

Dit onderwerp ligt op de kruising tussen multidimensionale meetkunde, grafentheorie en computer science, wat het interessant maakt voor iedereen met een brede interesse in wiskunde en technologie. Bovendien spreekt het zowel theoretische als praktische denkers aan: aan de ene kant biedt het diepe wiskundige structuren om te onderzoeken, aan de andere kant levert het algoritmische methoden op met tastbare

toepassingen.

Met dit profielwerkstuk willen we op een toegankelijke wijze inzicht bieden in hogere dimensies en aantonen hoe de abstracte wiskunde van polytopen en netstructuren intuïtief kan worden begrepen. We ontwerpen een algoritme dat, ongeacht de vorm van een polytoop, deze naar een lagere dimensie kan ontwikkelen. Deze methode wordt onder meer getest op bekende voorbeelden zoals de n-kubus, en we verkennen de diverse praktische toepassingen. Zó maken we abstracte wiskunde niet alleen inzichtelijk, maar benutten we tevens de educatieve en visuele kracht van dit fascinerende onderwerp.

1.3 Opbouw

In de komende hoofdstukken zullen we een systematische benadering volgen om ons onderzoek naar het ontwikkelen van hogere-dimensionale polytopen te presenteren.

Het werk begint met een theoretisch kader in Deel I (hoofdstukken 2-5), waarin we dieper ingaan op de wiskundige achtergrond van dimensies en polytopen. We behandelen achtereenvolgens euclidische ruimtes, de fundamentele eigenschappen van polytopen, netstructuren en essentiële concepten uit de grafentheorie. Om de lezer vertrouwd te maken met deze abstracte concepten, sluiten we Deel I af met een concrete casestudy over de n-kubus in hoofdstuk 6. Hierin onderzoeken we onder meer de recursieve definitie van de n-kubus, representeren we deze als een systeem van lineaire ongelijkheden, en analyseren we de structurele elementen ervan.

Deel II (hoofdstukken 7-11) presenteert onze eigen methode voor het ontwikkelen van polytopen. Na een inleidend hoofdstuk over de methode, beschrijven we hoe een polytoop kan worden gerepresenteerd als een graaf, hoe een optimale spanning tree gevonden kan worden, en ten slotte hoe deze tree kan worden omgezet naar een netstructuur. Om de praktische toepassing van onze methode te illustreren, bevat hoofdstuk 11 een uitgebreide casestudy waarin we zowel de 3-kubus als de complexere 7-kubus ontwikkelen.

In Deel III reflecteren we op ons onderzoek, bespreken we de beperkingen van onze benadering, en verkennen we mogelijke toepassingen in de praktijk. Daarnaast formuleren we vijf openstaande vraagstukken die aansluiten bij ons werk, maar waarop nog geen definitieve antwoorden zijn gevonden. Deze vraagstukken worden verder uitgewerkt in bijlage C.

Tot slot bevat dit werk een uitgebreide verklarende woordenlijst die de lezer helpt bij het begrijpen van de gebruikte terminologie, en diverse bijlagen met technische details en suggesties voor verder onderzoek.

Deel I

Theoretisch kader

Hoofdstuk 2

Euclidische ruimtes

Binnen ons onderzoek focussen we ons slechts op één definitie van n-dimensionale ruimtes, namelijk de euclidische ruimtes. In tegenstelling tot bijvoorbeeld een Riemann-ruimte of de 4D-ruimtetijd in de relativiteitstheorie, bevat deze bepaalde geometrische eigenschappen die essentieel zijn voor de meetkunde.

Figuur 2.1: Euclides

De euclidische ruimte vindt zijn oorsprong bij de Griekse wiskundige Euclides van Alexandrië, die rond 300 v.Chr. in zijn werk *The Thirteen Books of Euclid's Elements* de basis legde voor vlakke en 3-dimensionale meetkunde. Dit werk is een van de meest invloedrijke boeken in de wiskundige geschiedenis en wordt vaak genoemd als het op één na meest gepubliceerde boek na de Bijbel [42]. In de 19e eeuw breidden wiskundigen de euclidische ruimte uit naar hogere dimensies, wat leidde tot het moderne begrip van n-dimensionale euclidische ruimtes [5].

Hoewel het definiëren van een euclidische ruimte buiten de focus van dit theoretisch kader valt, is het belangrijk een paar van haar eigenschappen expliciet te benoemen, omdat deze later tijdens het opstellen van de methode van pas komen.

2.1 Vertexen en coördinaten

Het definiëren van een polytoop in een n-dimensionale ruimte is een van de essentieelste onderdelen van dit profielwerkstuk en wordt beter bekeken in hoofdstuk 3. Echter is het belangrijk om eerst de basis van elke polytoop beter te bekijken: de vertex. Een vertex (meervoud: vertexen) is een hoekpunt of een specifiek punt dat essentieel is voor de structuur van een polytoop. Later in dit werkstuk zullen we laten zien hoe een verzameling van deze vertexen gebruikt kan worden om een polytoop te definiëren. In deze paragraaf bekijken we voorbereidend hoe één vertex

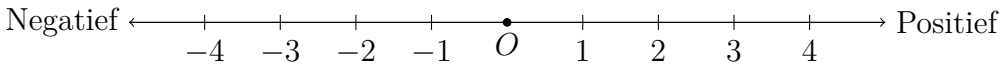
gepositioneerd kan worden in een n-dimensionale ruimte.

Om in een euclidische ruimte een vertex aan te geven, wordt het cartesisch coördinatenstelsel gebruikt. Dit systeem, bedacht door René Descartes en omschreven in zijn publicaties *Discours de la Méthode* en *La géométrie*, vormt de basis voor de beschrijving van posities en verhoudingen in de meetkunde. Een cartesisch coördinatenstelsel werkt door elke locatie in de ruimte te beschrijven aan de hand van een reeks getallen, de coördinaten, die de afstanden tot vastgelegde assen weergeven.

Het cartesisch coördinatenstelsel maakt het niet alleen mogelijk om punten nauwkeurig te lokaliseren, maar biedt ook de basis voor bewerkingen zoals transformaties, afstanden en hoeken tussen punten. Hierdoor biedt dit systeem een consistent kader om complexe structuren in een euclidische ruimte te definiëren en te manipuleren.

2.1.1 Getallenlijn

De eenvoudigste versie van het cartesisch coördinatenstelsel is de 1-dimensionale variant: een getallenlijn (fig. 2.2). Begin met een lijn en markeer hierop een vertex O, deze vertex wordt ook wel de oorsprong genoemd. Kies vervolgens een eenheid van lengte en aan welke kant zich de positieve getallen bevinden.



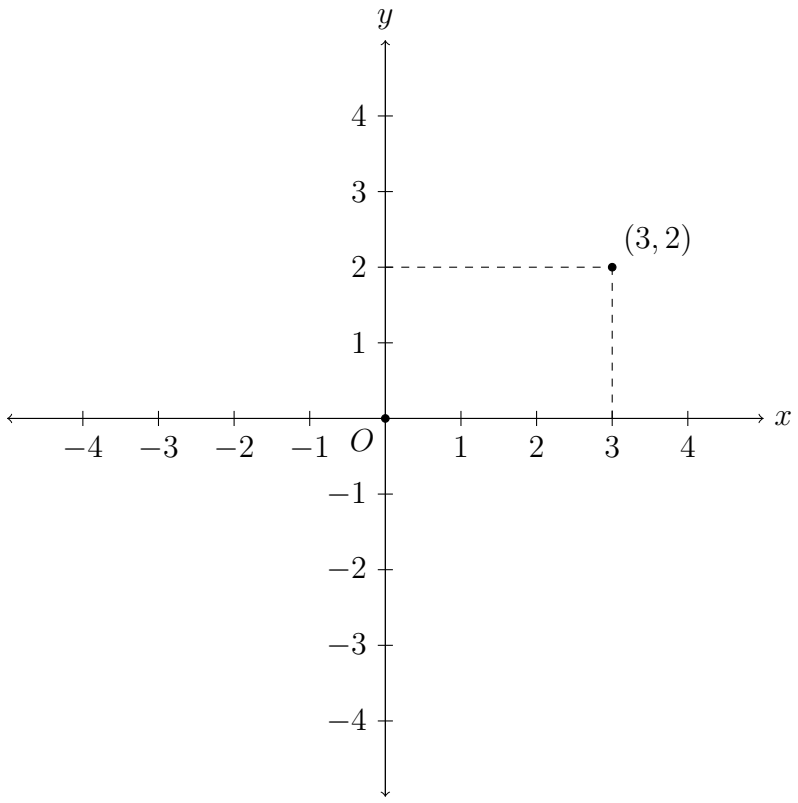
Figuur 2.2: Een getallenlijn

Op deze coördinatenlijn heeft elk reëel getal $x \in \mathbb{R}$ een unieke positie. Omgekeerd kan elke vertex op de lijn geïnterpreteerd worden door een getal x. Omdat elke vertex op de lijn gerepresenteerd kan worden door één enkel getal, spreken we hier van één dimensie.

Belangrijk is dat deze 1-dimensionale getallenlijn niet alleen gehele getallen, zoals 3 of -7, maar ook irrationele getallen, zoals π en $\sqrt{2}$, bevat. Dit betekent dat de lijn een volledige en continue representatie vormt van alle reële getallen, waarbij elk punt een exacte locatie heeft, ongeacht hoe klein of complex de waarde is.

2.1.2 Coördinatenvlak

Een cartesisch coördinatenvlak is te zien als twee getallenlijnen die loodrecht op elkaar staan. In een coördinatenvlak wordt de oorsprong O bepaald door het kruispunt tussen de getallenlijnen, die formeel de assen worden genoemd. Deze assen krijgen vaak de namen x en y. Door een eenheid van lengte te kiezen, die op beide assen gelijk is, is elke vertex in het vlak te beschrijven met getallen.



Figuur 2.3: Een cartesisch, tweedimensionaal assenstelsel

Zoals te zien is in fig. 2.3, kan een vertex[1] v in het vlak beschreven worden met twee getallen. Het eerste getal geeft aan waar de vertex zich bevindt op de x-as (horizontaal), terwijl het tweede getal de positie op de y-as aangeeft (verticaal). De getallen x en y zijn de coördinaten van vertex v, en worden geschreven als

$$v = (x, y)$$

Dit coördinaat is een geordende n-tuple. Dit is een reeks van n elementen, waarbij de volgorde van deze elementen van belang is. De vertexen $(3, 2)$ en $(2, 3)$ zijn immers niet hetzelfde!

Omdat een vertex in dit geval door twee getallen wordt omschreven, spreken we van een 2-tuple. Elke vertex in het 2-dimensionale cartesisch coördinatenstelsel kan daarom formeel worden uitgedrukt als een element van de verzameling $\mathbb{R}^2$. Deze verzameling $\mathbb{R}^2$ staat voor alle mogelijke geordende 2-paren van reële getallen. Dit betekent dat als $x \in \mathbb{R}$ en $y \in \mathbb{R}$, dan kan elke vertex in het coördinatenvlak worden weergegeven als een tuple (x, y).

Het is hierbij belangrijk om te begrijpen dat $\mathbb{R}^2$ gelijk is aan het cartesisch product van $\mathbb{R}$ met zichzelf: $\mathbb{R} \times \mathbb{R} = \mathbb{R}^2$. Dit betekent dat de verzameling $\mathbb{R}^2$ bestaat

[1] Kleine letters worden gebruikt voor vertexen en vectoren om de notatie consistent te houden, in tegenstelling tot sommige werken waar voor een vertex hoofdletters worden gebruikt.

uit alle mogelijke combinaties van twee reële getallen, één voor de x-coördinaat en één voor de y-coördinaat. Elke unieke geordende 2-tuple in $\mathbb{R}^2$ komt dus overeen met een unieke vertex in het coördinatenvlak, en omgekeerd is elke vertex in dit vlak te omschrijven met zo'n geordende 2-tuple.

Met andere woorden: het coördinatenvlak is niets anders dan de verzameling van alle mogelijke paren van reële getallen en dit wordt uitgedrukt als $\mathbb{R}^2$.

Vectorrepresentatie van een punt

In de context van een cartesisch coördinatenstelsel kan elke vertex v niet alleen worden beschreven met coördinaten, maar ook door een vector die naar dat punt wijst vanuit de oorsprong O. Een vector is een wiskundig object dat zowel een richting als een grootte heeft. In dit geval wordt de vector $\boldsymbol{v}$ die naar de vertex v wijst gedefinieerd als:

$$\boldsymbol{v} = \begin{pmatrix} x \\ y \end{pmatrix}$$

Hierbij vertegenwoordigen de coördinaten x en y de horizontale en verticale afstand van de oorsprong O naar de vertex v. Deze representatie is bijzonder nuttig omdat vectoren in de wiskunde en de natuurkunde vaak gemakkelijker te hanteren zijn dan coördinaten, omdat ze bepaalde basis-bewerkingen hebben. Een gedeelte van deze bewerkingen wordt verspreid door het verslag toegelicht.

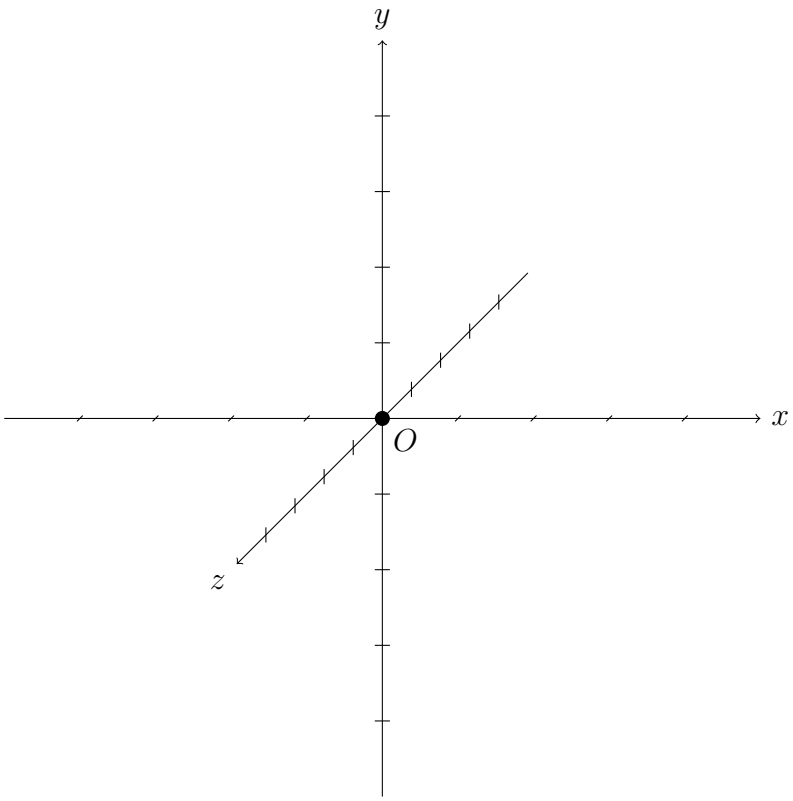
Opmerking 1. Op de middelbare school wordt als vectornotatie vaak $\vec{v}$ gebruikt, in de wetenschappelijke literatuur is het echter gebruikelijker om de dikgedrukte $\boldsymbol{v}$ te gebruiken. Om beter aan te sluiten bij bestaand onderzoek, zullen we in het vervolg slechts de tweede notatie gebruiken.

2.1.3 Driedimensionaal coördinatenstelsel

Door dit patroon te volgen, kunnen we relatief makkelijk een definitie geven aan de derde dimensie. Door een derde as loodrecht op de bestaande assen te plaatsen, breiden we het 2-dimensionale coördinatenvlak uit naar drie dimensies:

$$\mathbb{R}^3 = \mathbb{R}^2 \times \mathbb{R} = \mathbb{R} \times \mathbb{R} \times \mathbb{R}$$

Een vertex in een 3-dimensionale ruimte kan, het patroon volgend, dus geschreven worden als een geordende 3-tuple: $v = (x, y, z)$.



Figuur 2.4: Driedimensionaal coördinatenstelsel

Dit driedimensionale coördinatenstelsel is afgebeeld in fig. 2.4. Voor een alien die niet bekend is met de derde dimensie, zal dit er wellicht uit zien alsof de z-as diagonaal loopt. Elk mens zou deze alien er echter op kunnen wijzen dat dit niet het geval is. De z-as loopt naar achteren, maar omdat dit op tweedimensionaal papier is weergeven, lijkt de as schuin te lopen. Hij loopt niet schuin, hij loopt gewoon naar achteren.

2.1.4 Hogerdimensionaal coördinatenstelsel

Op deze manier is het mogelijk alle volgende dimensies een definitie te geven. Door een nieuwe as loodrecht op de bestaande assen te plaatsen, breiden we het stelsel uit naar een dimensie hoger. Toch zorgt dit weer voor nieuwe problemen, maar deze komen vooral door het bevattingsvermogen van de mens. Om deze reden maken we eerst een paar aanpassingen aan onze definitie.

Benaming van de assen

Het eerste probleem is een vrij praktische: we hebben maar 26 letters in het Latijnse alfabet, niet genoeg om *alle* assen een eigen letter te geven. Hoewel we dit kunnen oplossen door ook bijvoorbeeld het Grieks of Arabisch alfabet te gebruiken, lopen we ook hierbij weer tegen praktische problemen aan, bovendien is het nog steeds

geen oneindig aantal letters. Het is dus niet haalbaar aan elke as een eigen letter toe te kennen.

Om deze reden worden de assen genummerd. Hoewel dit wellicht voor de hand lijkt te liggen, kan dit aanleiding geven tot verwarring, en het is daarom essentieel dit expliciet te verduidelijken. De coördinaten van een vertex, aangeduid met x, in een n-dimensionale ruimte worden genoteerd door een geordende n-tuple:

$$x = (x_1, x_2, x_3, \ldots, x_n)$$

Hieruit wordt meteen het potentieel probleem zichtbaar: het punt zelf wordt aangeduid met de letter x, waardoor x niet langer refereert aan een van de coördinaatassen. Om onbegrijpelijke redenen is het gebruik van de letters x en y als twee punten, de algemeen geaccepteerde standaard bij het werken in hogere dimensies, wat bij veel mensen voor de nodige verwarring zorgt. Op de middelbare school wordt aangeleerd dat x en y assen zijn, terwijl dit nu juist plekken op de assen voorstellen. In dit voorbeeld zullen we echter voor consistentie de letter v blijven gebruiken, maar de notatie (x, y) als coördinaat zal de komende 96 bladzijden niet meer terugkomen.

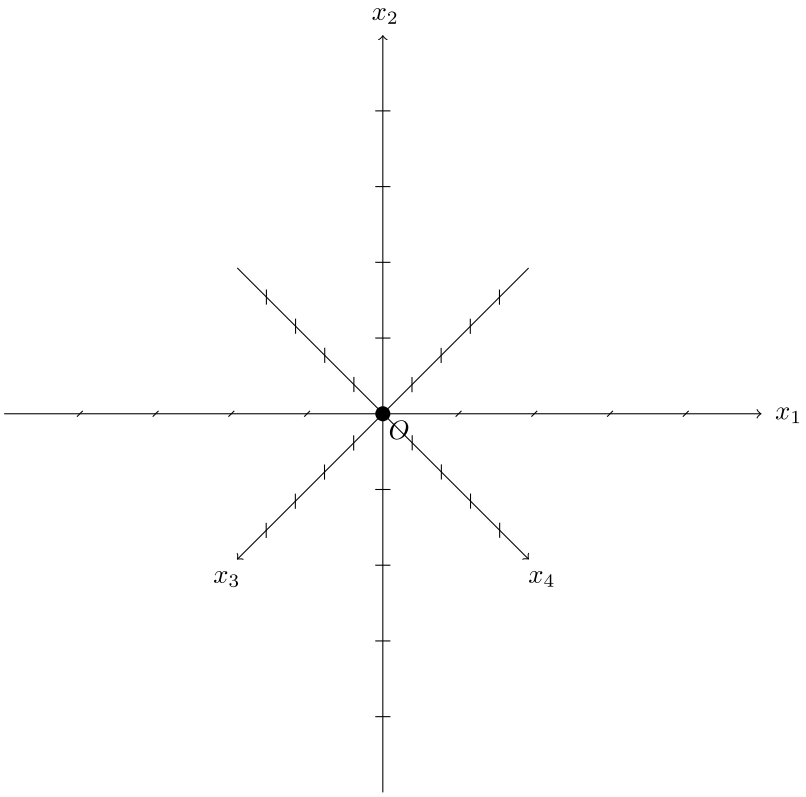
Visualisering van de vierde dimensie

Een complexer probleem is het visualiseren van de vierde dimensie. Voor ons, wezens uit een driedimensionaal universum, is het lastig te bevatten hoe er meer dimensies, loodrecht op onze drie dimensies kunnen bestaan. Sterker nog, stel dat we in een vierdimensionaal universum leefden, is het onmogelijk hier zelf achter te komen [41], omdat onze zintuigen en ons begrip van ruimte slechts in drie dimensies functioneren.

De beste manier om met hogere dimensies te werken, is door ze *niet* te visualiseren. Door op een systematische manier over dimensies te denken, zonder het proberen in beeld te brengen, kan er een prima intuïtie voor worden ontwikkeld. Geen intuïtie voor hoe een n-dimensionale ruimte eruitziet, maar wel voor hoe een n-dimensionale ruimte verschilt van een $(n-1)$-dimensionale ruimte. Dit is genoeg.

Een goed boek om hiermee te helpen, is *Flatland* door Edwin Abbott. Dit boek verkent een hypothetische wereld die zich in twee dimensies afspeelt, waardoor goed te zien is hoe onze 3-dimensionale wereld hiervan afwijkt.

In fig. 2.5 hebben we toch een poging gedaan om een vierdimensionaal coordinatenstelsel te visualiseren. Dit ziet er alleen totaal niet logisch uit. Wat gebeurt er met die schuine assen? Een alien die de vierde dimensie gewend is zal simpelweg zeggen: “De as loopt niet schuin, hij loopt gewoon naar achteren.” Diezelfde alien zou wanneer hij een illustratie van de vijfde dimensie bekijkt zich echter precies hetzelfde afvragen: wat doet die schuine as daar?



Figuur 2.5: Vierdimensionaal coördinatenstelsel

Benaming van het coördinatenstelsel

Het meest voorspelbare binnen het werken met hoge dimensies, is hun benaming. Deze blijven de vorm $\mathbb{R}^n$ volgen, waarbij n staat voor de dimensie, oftewel het aantal assen dat haaks op elkaar staan. De Euclidische vierde dimensie wordt dus omschreven met $\mathbb{R}^4$.

Algemene regels

Concluderend, wordt een vertex v in een cartesisch n-dimensionaal coördinatenstelsel omschreven door een geordende n-tuple of een n-dimensionale vector:

$$v = (v_1, v_2, \ldots, v_n) \quad \text{of} \quad \boldsymbol{v} = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix} \tag{2.1}$$

De n dimensies houden in dat er binnen het coördinatenstelsel n assen zijn. Cartesisch verwijst naar het feit dat deze assen allen elkaar in de oorsprong O snijden en elkaar raken op een hoek van 90 graden.

2.2 Meetkundige eigenschappen

Een euclidische ruimte bezit diverse meetkundige eigenschappen die in elke dimensie gelden. De volgende voorbeelden illustreren enkele van deze eigenschappen en bieden inzicht in het rekenen binnen euclidische ruimtes. Dit is echter geen uitputtende lijst van alle eigenschappen die een euclidische ruimte definiëren.

2.2.1 Afstand tussen punten

Binnen elke euclidische ruimte is er een vaste, kortste afstand tussen twee punten. Deze afstand wordt gegeven door de euclidische afstandsformule.

Definitie 1. De *Euclidische afstand* d tussen twee punten $x = (x_1, x_2, \ldots, x_n)$ en $y = (y_1, y_2, \ldots, y_n)$ in de euclidische ruimte $\mathbb{R}^n$ wordt gedefinieerd als:

$$d = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \cdots + (x_n - y_n)^2} \tag{2.2}$$

Als voorbeeld wordt in de 2-dimensionale euclidische ruimte $\mathbb{R}^2$ de afstand d tussen de punten $x = (x_1, x_2)$ en $y = (y_1, y_2)$ als volgt berekend:

$$d = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}$$

Deze formule komt direct overeen met de *stelling van Pythagoras* wanneer x en y de uiteinden van een rechthoekige driehoek vormen. Volgens deze stelling is de lengte van de schuine zijde c van een rechthoekige driehoek gelijk aan:

$$a^2 + b^2 = c^2 \implies c = \sqrt{a^2 + b^2}$$

waarbij a en b de lengtes van de rechthoekszijden zijn. Deze formule is identiek aan de 2-dimensionale afstandsformule.

2.2.2 Inproduct

De euclidische ruimte heeft ook een gedefinieerd inwendig product, vaak het scalair product genoemd, dat een maat is voor zowel de grootte van de projectie van de ene vector op de andere als voor de hoek tussen twee vectoren.

In een ruimte $\mathbb{R}^n$ is het inproduct van twee vectoren $\boldsymbol{u}$ en $\boldsymbol{v}$ gedefinieerd als:

$$\boldsymbol{u} \cdot \boldsymbol{v} = u_1 v_1 + u_2 v_2 + \cdots + u_n v_n \tag{2.3}$$

Dit resulteert in een reëel getal, het scalair product van $\boldsymbol{u}$ en $\boldsymbol{v}$. Het inproduct kan

ook worden uitgedrukt in termen van de hoek θ tussen twee vectoren $\boldsymbol{u}$ en $\boldsymbol{v}$:

$$\boldsymbol{u} \cdot \boldsymbol{v} = \|\boldsymbol{u}\| \cdot \|\boldsymbol{v}\| \cdot \cos\theta \tag{2.4}$$

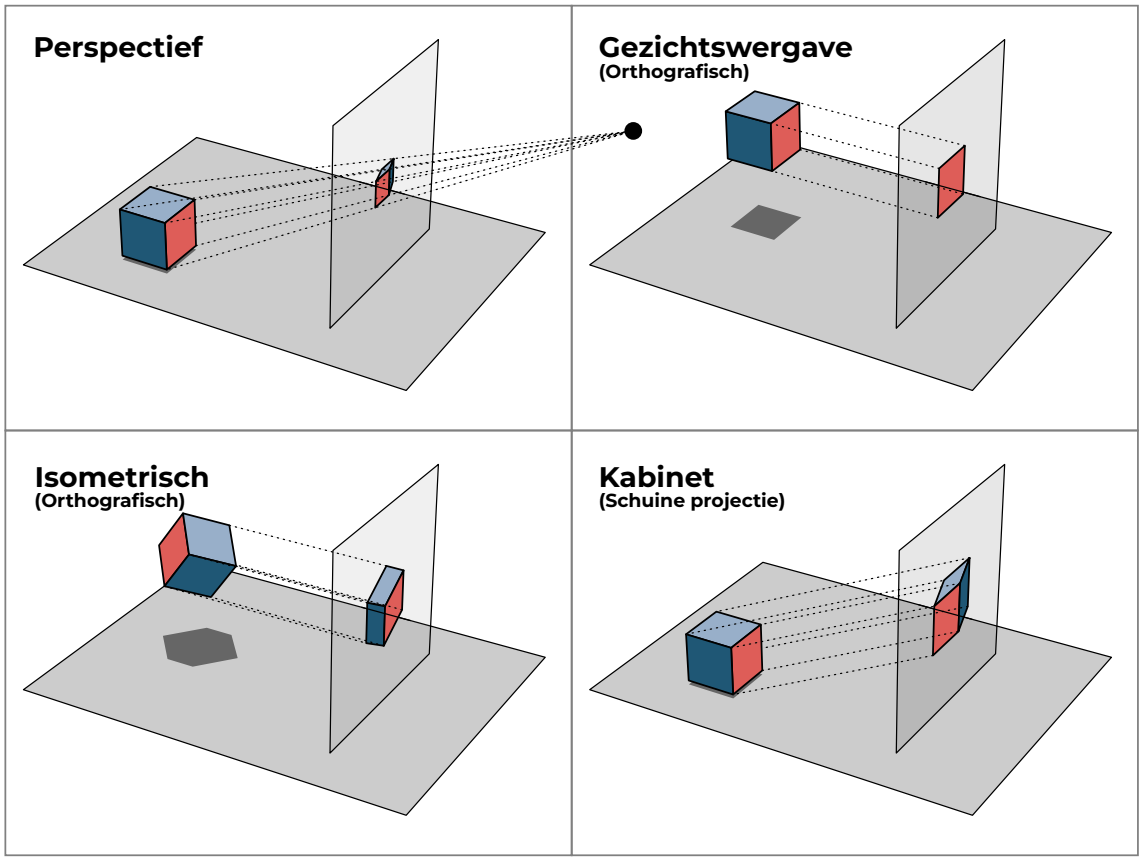
waarbij $\|\boldsymbol{u}\|$ en $\|\boldsymbol{v}\|$ de lengtes (of normen) van de vectoren zijn, gegeven door:

$$\|\boldsymbol{u}\| = \sqrt{u_1^2 + u_2^2 + \cdots + u_n^2} = \sqrt{\boldsymbol{u} \cdot \boldsymbol{u}} \tag{2.5}$$

Deze formule lijkt qua vorm op verg. (2.2) en is hier ook van af te leiden. Dit is uitgewerkt in appendix A.

2.3 Projecties

Projecties zijn een fundamenteel concept binnen de meetkunde en lineaire algebra, waarmee punten uit een ruimte van een bepaalde dimensie naar een ruimte van lagere dimensie kunnen worden afgebeeld. Het projecteren van een object uit een hogere dimensie naar een lagere dimensie kan worden gezien als een soort 'schaduw' van dat object in een minder complexe ruimte. De meest gangbare projecties in de Euclidische meetkunde zijn orthogonale projecties en perspectiefprojecties.



Figuur 2.6: Een visualisatie van verschillende projectie-methoden (naar [37])

2.3.1 Orthogonale projectie

De orthogonale projectie is een veelgebruikte techniek in de euclidische meetkunde om een punt loodrecht op een lagere-dimensionale subruimte te plaatsen. Dit betekent dat het verschil tussen het originele punt en de projectie loodrecht staat op de subruimte.

Een eenvoudige manier om dit te doen is door bepaalde coördinaten weg te laten. Bijvoorbeeld, als een vertex

$$v = (v_1, v_2, v_3)$$

in $\mathbb{R}^3$ wordt geprojecteerd op een vlak dat door de eerste twee coördinaten wordt opgespannen, dan is de projectie:

$$v' = (v_1, v_2, 0).$$

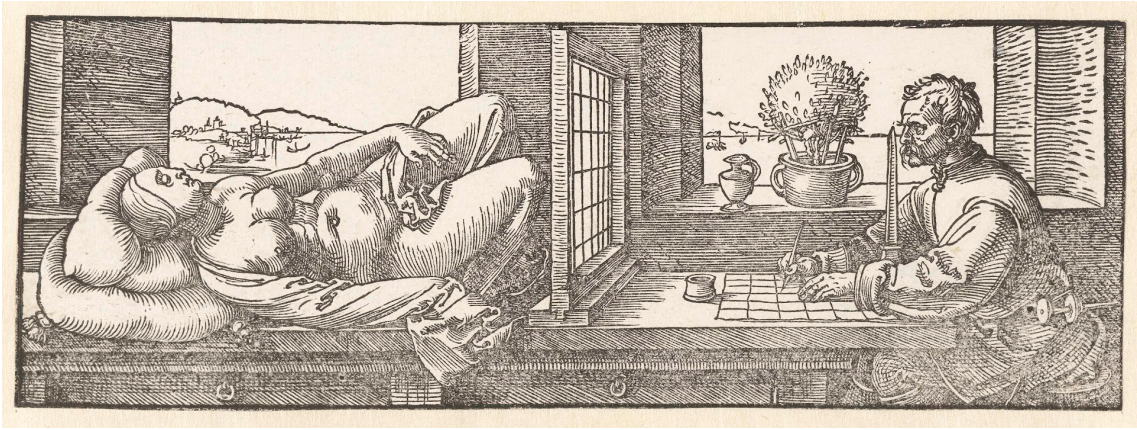
Als de subruimte niet precies langs de assen ligt, kan het inproduct worden gebruikt om te bepalen hoe het punt geprojecteerd moet worden. Hiermee kan een vertex op elke willekeurige subruimte worden geprojecteerd.

Orthogonale projectie helpt bij het eenvoudiger maken van gegevens uit hogere dimensies en wordt veel gebruikt in toepassingen zoals computergraphics en data-analyse. In deze methode blijven verhoudingen en afstanden gedeeltelijk behouden, wat het onderscheidt van een andere veelgebruikte techniek: de perspectiefprojectie.

2.3.2 Perspectiefprojectie

In contrast met de orthogonale projectie, behoudt de perspectiefprojectie niet de onderlinge afstanden van punten. Een perspectiefprojectie wordt vaak gebruikt in de grafische weergave en schilderkunst om diepte te suggereren. Hierbij worden punten van een hoger-dimensionaal object afgebeeld op een lagere-dimensionale ruimte door ze vanuit een vast gezichtspunt 'naar voren' te projecteren. Objecten die zich verder van het gezichtspunt bevinden, worden kleiner weergegeven dan objecten die dichterbij zijn, waardoor de illusie van diepte ontstaat.

De kunstenaar Albrecht Dürer (fig. 2.7) speelde een belangrijke rol in het onderzoeken en vastleggen van perspectieftechnieken in de renaissancekunst, waarmee hij een nauwkeurig systeem ontwikkelde om driedimensionale objecten realistisch op een plat vlak weer te geven. Deze methoden worden ook toegepast om 3-dimensionale objecten op een tweedimensionaal scherm te projecteren, zoals bij computergames en 3-dimensionale modellen. Hoewel perspectiefprojecties in de theoretische wiskunde weinig toepassingen hebben, vanwege de verandering in geometrische eigenschappen, geven ze de kijker een meer realistische perceptie van de vorm en diepte van 3-dimensionale objecten.



Figuur 2.7: Albrecht Dürer (rechts) tekent met een raster in perspectief

2.3.3 Complexe projecties

Naast orthogonale en perspectiefprojecties bestaan er veel andere vormen van projectie, die in verschillende toepassingen worden gebruikt. Hoewel veel van deze niet centraal staan in dit onderzoek, is het belangrijk om te benoemen dat projecties op verschillende manieren kunnen worden toegepast, afhankelijk van de context en het doel.

Een voorbeeld hiervan, en tevens het onderwerp van dit onderzoek, is het ontwikkelen van een polytoop, wat kan worden beschouwd als een complexe vorm van projectie. Hierbij wordt een object uit een hogere dimensie weergegeven in een lagere dimensie, terwijl bepaalde structurele eigenschappen behouden blijven. Dit proces speelt een belangrijke rol bij het visualiseren en analyseren van hogere-dimensionale objecten. Meer over netstructuren bevindt zich in hoofdstuk 4.

Hoofdstuk 3

Polytopen

Een polytoop is simpel gezegd een geometrisch object dat in een willekeurig aantal dimensies kan bestaan. In het dagelijkse leven kom je polytopen overal tegen, dit zijn polytopen in de derde dimensie, denk aan kubussen of piramides. Een driedimensionale polytoop wordt ook wel een veelvlak of polyhedron genoemd. Ook bestaan er polytopen in de nulde dimensie (een enkele vertex), de eerste dimensie (een lijnsegment), en zo door tot de n-de dimensie, die ontelbaar groot kan zijn.

In de eerste, tweede en derde dimensie zijn polytopen nog simpel te begrijpen, maar als je gaat kijken naar polytopen in de hogere dimensies (4D, 5D, etc.) wordt het echter complexer. Polytopen in deze hogere dimensies kunnen wij als wezens in een 3-dimensionaal universum niet voorstellen, maar we kunnen ze wel theoretisch en wiskundig definiëren.

Het woord polytoop is afkomstig uit het Grieks en bestaat uit twee delen: $\pi o \lambda \upsilon$ (poly) en $\tau o \pi o \varsigma$ (topos). Het eerste deel betekent "veel", terwijl het tweede deel "plaats" of "ruimte" betekent. In de wiskunde verwijst een polytoop naar een algemene vorm met "veel ruimtes" lees "veel dimensies".

3.1 Definitie

Een polytoop is een geometrisch object dat in verschillende dimensies kan voorkomen. Wiskundig gezien is een polytoop de omhulling van een eindige verzameling punten in een euclidische ruimte. Afhankelijk van het aantal dimensies van de polytoop kunnen we enkele voorbeelden geven:

- 0-dimensionaal: een enkele vertex of punt;
- 1-dimensionaal: een lijnsegment, gevormd door twee verbonden vertexen;
- 2-dimensionaal: een veelhoek, zoals een driehoek of een vierkant;
- 3-dimensionaal: een veelvlak, zoals een kubus of een tetraëder;

- 4-dimensionaal en hoger: objecten zoals de tesseract (4-kubus) of simplex.

In algemene termen is een n-dimensionale polytoop de verzameling oplossingen van een systeem van lineaire ongelijkheden:

$$A\boldsymbol{x} \leq \boldsymbol{b}, \tag{3.1}$$

waarbij A een matrix is en $\boldsymbol{x}$ en $\boldsymbol{b}$ vectoren zijn. Dit betekent dat een polytoop wordt gedefinieerd als alle punten die voldoen aan deze voorwaarden binnen een begrensde ruimte. Voorbeelden van deze notatie bevinden zich in paragraaf 6.2.

3.1.1 Facetten van een polytoop

Een polytoop kan ook worden beschreven in termen van zijn facetten. Facetten zijn de $(n-1)$-dimensionale 'randen' die de grenzen van de polytoop vormen. Ze zijn dus altijd één dimensie lager dan de polytoop. Enkele voorbeelden zijn:

- Een lijnsegment heeft 2 vertexen (0-dimensionale punten) als facetten;
- Een vijfhoek heeft 5 lijnen (1-dimensionale randen) als facetten;
- Een kubus heeft 6 vierkanten (2-dimensionale vlakken) als facetten.

3.2 Elementen van een polytoop

Een polytoop bestaat uit meerdere elementen, afhankelijk van de dimensie. Zo bestaat een driehoek uit 3 vertexen (0-dimensionaal), 3 lijnstukken (1-dimensionaal), en 1 vlak (2-dimensionaal). Deze bouwstenen worden aangeduid als k-dimensionale elementen van de polytoop. Voor een n-dimensionale polytoop P_n gelden de volgende elementen:

- F_0: vertex ($k = 0$);
- F_1: lijnstuk ($k = 1$);
- F_2: vlak ($k = 2$);
- F_3: cel ($k = 3$);
- F_{n-1}: facet ($k = n - 1$).

Hierbij geeft F_k de verzameling van alle k-dimensionale elementen in de polytoop weer. Met andere woorden:

$$F_k = \{f \mid f \text{ is een } k\text{-dimensionaal facet van de polytoop } P_n\}. \tag{3.2}$$

Elk k-dimensionale element is een deelverzameling van de polytoop en vormt een belangrijke structuur binnen de hiërarchie van elementen. Zo bevat:

- F_0: alle punten (vertexen) van de polytoop;
- F_1: alle lijnstukken die worden gevormd door de verbinding van twee vertexen;
- F_2: alle vlakken die worden begrensd door F_1-lijnstukken;
- F_3: alle cellen die worden begrensd door F_2-vlakken;
- F_{n-1}: de facetten die de rand van de polytoop vormen;
- F_n: de complete polytoop.

Door deze hiërarchie kunnen we een polytoop beschrijven als een verzameling facetten van verschillende dimensies, waarbij de structuur van hogere-dimensionale facetten is opgebouwd uit de lagere-dimensionale facetten.

Opmerking 2. In sommige onderzoeken wordt de term facet gebruikt voor elk element met een dimensie lager dan die van de polytoop. In dit onderzoek richten we ons echter voornamelijk op elementen met $k = n - 1$. Daarom definiëren we een facet als een element waarvan de dimensie precies één lager is dan die van de polytoop. Om een term te hebben voor een object met een dimensie lager dan de polytoop (maar niet noodzakelijkerwijs slechts één lager), introduceren we de term element. Dit stelt ons in staat op een makkelijke manier te praten over de verschillende onderdelen van een polytoop en hun relaties.

3.3 4D en hoger

Polytopen in 4D en hogere dimensies kunnen wij ons als 3D-wezens moeilijk voorstellen, maar theoretisch kunnen we ze beschrijven en analyseren. Zo wordt de vierdimensionale versie van een kubus een tesseract genoemd (zie hoofdstuk 6). Dit is een voorbeeld van een polytoop die wordt opgebouwd door de relaties tussen 3D-objecten. In hogere dimensies, zoals 5D en 6D, ontstaan er nog complexere vormen die we alleen wiskundig kunnen begrijpen, omdat ze niet direct te visualiseren zijn in onze driedimensionale wereld.

Voorbeeld: algemene 4D polytoop

Een algemeen voorbeeld van een 4D polytoop is een object dat kan worden beschreven door een set lineaire ongelijkheden. Elk punt dat aan deze ongelijkheden

voldoet, bevindt zich in de polytoop. Al deze punten samen vormen de gehele polytoop. Dat deze ongelijkheden lineair zijn, zorgt ervoor dat de polytoop altijd rechte ribben heeft.

Als voorbeeld is een vierdimensionale kubus de verzameling van punten in $\mathbb{R}^4$ die voldoen aan de volgende lineaire ongelijkheden:

$$\begin{cases} -1 \leq x_1 \leq 1 \\ -1 \leq x_2 \leq 1 \\ -1 \leq x_3 \leq 1 \\ -1 \leq x_4 \leq 1 \end{cases} \tag{3.3}$$

Dit is een voorbeeld van hoe lineaire ongelijkheden kunnen worden gebruikt om een polytoop in een hogere dimensie te beschrijven. De specifieke eigenschappen en het aantal facetten van zulke objecten variëren afhankelijk van de dimensie, maar het algemene principe blijft hetzelfde.

3.4 Eigenschappen van polytopen

Polytopen in hogere dimensies hebben vaak bijzondere (symmetrische) eigenschappen die kunnen worden bestudeerd met behulp van groepentheorie. De symmetrie van een polytoop hangt nauw samen met de onderlinge relaties tussen zijn facetten, randen en hoekpunten. Wiskundig gezien kan een polytoop ook worden gekarakteriseerd door zijn Euler-karakteristiek, die de relatie tussen het aantal hoekpunten, ribben en facetten beschrijft.

De polytoopformule van Euler is een wiskundige regel die een verband legt tussen de bouwstenen van polytopen in verschillende dimensies. De formule veralgemeent het verband tussen deze elementen naar hogere dimensies en is een uitbreiding van de klassieke formule voor 2D-vormen.

3.4.1 De klassieke formule: 2D en 3D

In 3D (veelvlakken zoals een kubus) geeft Euler ons het volgende basisresultaat:

$$\#F_0 - \#F_1 + \#F_2 = 2 \tag{3.4}$$

Hierbij staat:

- $\#F_0$ voor het aantal hoekpunten,
- $\#F_1$ voor het aantal randen,

- $\#F_2$ voor het aantal vlakken.

Dit resultaat geldt voor convexe vormen, zoals een kubus of een tetraëder.
(In 2D is het resultaat vaak 1, omdat er geen 'buitenste' volume is zoals in 3D.)

Voorbeeld 1 (Kubus)**.** De polytoopformule van Euler ziet er voor een driedimensionale kubus als volgt uit:

- **Hoekpunten**: $\#F_0 = 8$ (elk hoekpunt van de kubus),
- **Randen**: $\#F_1 = 12$ (elke rand van de kubus),
- **Vlakken**: $\#F_2 = 6$ (elk vlak van de kubus).

De formule wordt dan:

$$\#F_0 - \#F_1 + \#F_2 = 8 - 12 + 6 = 2$$

3.4.2 Uitbreiding naar hogere dimensies

In hogere dimensies wordt de formule veralgemeend. Voor een n-dimensionale polytoop wordt het verband:

$$\#F_0 - \#F_1 + \#F_2 - \cdots + (-1)^n \cdot \#F_n = \chi \tag{3.5}$$

Hierbij is:

- $\#F_n$ het aantal n-dimensionale facetten,
- $(-1)^n$ de factor die afwisselend optellen en aftrekken regelt,
- χ de *Euler-karakteristiek*.

Voorbeeld 2 (Tesseract)**.** Een tesseract is een 4D-veralgemening van de kubus, meer hierover is te vinden in hoofdstuk 6. De polytoopformule van Euler ziet er hiervoor als volgt uit:

- **Hoekpunten**: $\#F_0 = 16$,
- **Randen**: $\#F_1 = 32$,
- **Vlakken**: $\#F_2 = 24$ (vierkanten),
- **Facetten**: $\#F_3 = 8$ (kubussen).

De formule wordt dan:

$$\#F_0 - \#F_1 + \#F_2 - \#F_3 = 16 - 32 + 24 - 8 = 0$$

3.4.3 Waarom is dit belangrijk?

1. **Structuur begrijpen**: De formule helpt ons te begrijpen hoe de verschillende elementen (hoekpunten, randen, vlakken, etc.) van een figuur in elkaar grijpen, zelfs in hoge dimensies.

2. **Controlemechanisme**: Bij het bouwen of bestuderen van complexe geometrische vormen kun je de formule gebruiken om te controleren of je de juiste aantallen hebt.

3. **Generaliseerbaarheid**: De formule werkt consistent in alle dimensies, van 2D tot 4D en verder, waardoor het een universeel hulpmiddel is in de geometrie.

4. **Rekenkundige eenvoud**: Door dit verband hoef je niet altijd alle elementen afzonderlijk te tellen; je kunt sommige aantallen afleiden door de formule te gebruiken.

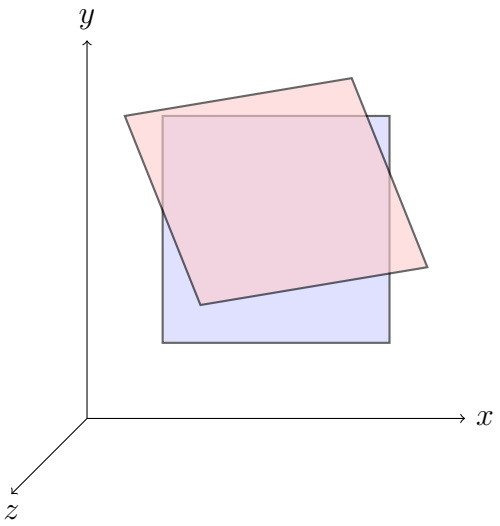
3.5 Polytopen in een affiene deelruimte

Een polytoop in een n-dimensionale euclidische ruimte kan zich bevinden in een affiene deelruimte van een lagere dimensie. Een affiene ruimte lijkt op een euclidische ruimte, maar heeft geen vaste oorsprong (nulpunt). Dankzij deze eigenschap kan de deelruimte onbeperkt worden verschoven en gedraaid zonder dat de onderliggende geometrische structuur verandert.

Een voorbeeld van een polytoop in een affiene deelruimte is een vierkant dat zich in een 3-dimensionale ruimte bevindt. Hoewel het vierkant een tweedimensionaal object is, kan het in een 2-dimensionale affiene deelruimte van de 3-dimensionale ruimte liggen. Door de affiene deelruimte te verschuiven of te draaien, blijft het vierkant in een deelruimte, maar verandert de positie of oriëntatie binnen de 3-dimensionale ruimte.

Concreet betekent dit dat het vierkant bijvoorbeeld loodrecht op een van de coördinaatassen kan liggen, of onder een willekeurige hoek in de ruimte. In beide gevallen blijft het een plat, tweedimensionaal object, maar het ligt in verschillende affiene deelruimtes van de 3-dimensionale ruimte. Dit is een belangrijk concept in de studie van polytopen omdat het laat zien hoe objecten van lagere dimensies zich kunnen manifesteren in hogere-dimensionale ruimtes.

In figuur 3.1 zien we twee voorbeelden van een vierkant in een 3-dimensionale ruimte. Het blauwe vierkant ligt loodrecht op de z-as en bevindt zich in een affiene deelruimte die parallel is aan het xy-vlak. Het rode vierkant is hetzelfde object, maar het is verschoven en gedraaid, wat laat zien hoe de affiene deelruimte kan veranderen. Ondanks deze veranderingen blijft het object zelf altijd tweedimensionaal. Deze



Figuur 3.1: Twee vierkanten in een 3-dimensionale ruimte

transformaties kunnen ook de andere kant uit: het rode vierkant kan verplaatst worden om de blauwe te krijgen.

Voor elk hoekpunt $V_i = (x, y, z)$ in het blauwe vierkant geldt dat $z = 0$. Om deze reden ligt dit vierkant, en alle verplaatsingen ervan, in een 2-dimensionale deelruimte.

3.5.1 Berekenen van de dimensionaliteit van een polytoop

De dimensionaliteit van een polytoop in een n-dimensionale ruimte geeft aan in welke affiene deelruimte de polytoop ligt. Hetzelfde is andersom waar. Helaas is het berekenen van de dimensionaliteit van een affiene deelruimte complex. Het is lastig uit te leggen wat er gebeurt, en ook wij begrijpen niet volledig waarom bepaalde stappen worden gezet.

In appendix B staat stap voor stap beschreven hoe deze dimensionaliteit te berekenen is, maar voor nu houden we het op de volgende samenvatting: om de dimensionaliteit van een polytoop te vinden, moet je proberen de hele polytoop in een ruimte te stoppen met een zo laag mogelijke dimensie.

Dit is ook te zien als het stoppen van een vorm in een n-dimensionale polytoop (denk aan een vierkant en een kubus) met een zo laag mogelijke dimensie. Het zal je bijvoorbeeld nooit lukken om een bowlingbal in een 2-dimensionaal vel papier te stoppen, hoe je de bowlingbal ook draait. Zelfs met een eindeloos groot papier kan het niet, je komt simpelweg een dimensie te kort.

Probeer je diezelfde bowlingbal in een 3-dimensionaal object te stoppen, kan dat wel (zo lang de afmeting van dat object groter is dan de bowlingbal). Om deze reden, is een bowlingbal 3-dimensionaal; de kleinste affiene deelruimte waar hij in past, is namelijk ook 3-dimensionaal.

Hoofdstuk 4

Netstructuren

Een belangrijk concept bij het bestuderen van hogere-dimensionale polytopen is het uitvouwen ervan naar een lager-dimensionale netstructuur. Dit proces, ook wel *ontplooiing* of *ontwikkeling* genoemd, stelt ons in staat om complexe polytopen te visualiseren en te analyseren door ze af te beelden in een ruimte van één dimensie lager.

4.1 Definitie en eigenschappen

Definitie 2. Een netstructuur van een n-dimensionale polytoop is een $(n-1)$-dimensionale representatie die ontstaat door het 'opensnijden' en 'uitvouwen' van de polytoop langs bepaalde randen, zodanig dat alle facetten behouden blijven en hun onderlinge verbindingen duidelijk zijn.

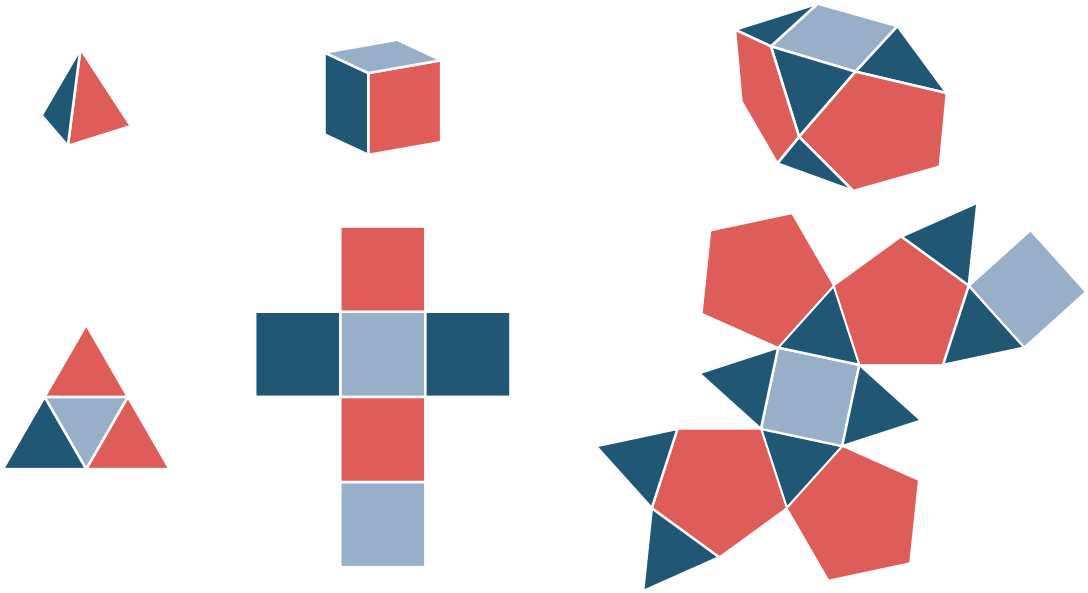
Figuur 4.1 toont drie voorbeelden van netstructuren. Links zien we een eenvoudige tetraëder en het bijbehorende net, in het midden een kubus met een van zijn mogelijke netstructuren, en rechts een complexere bilunabirotunda met zijn net. Deze illustraties laten zien hoe de oorspronkelijke polytopen uitvouwen naar platte representaties.

4.1.1 Behoud van eigenschappen

Bij de transformatie van een polytoop naar zijn netstructuur blijven verschillende fundamentele eigenschappen behouden. Deze eigenschappen zijn essentieel voor zowel het theoretische begrip als de praktische toepasbaarheid van netstructuren.

Bij de transformatie van een polytoop naar zijn netstructuur blijven de volgende eigenschappen behouden:

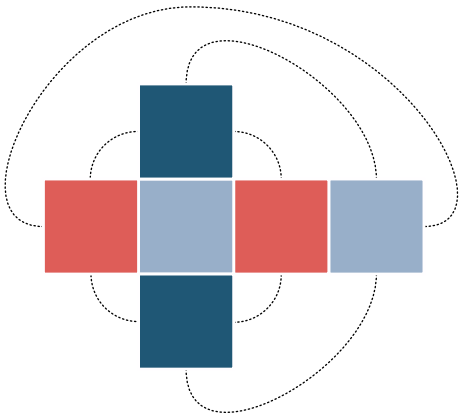
- De *hyperoppervlakte* van elk facet blijft ongewijzigd;
- De lengte van elke ribbe (1-dimensionale rand) blijft gelijk;



Figuur 4.1: Voorbeelden van netstructuren: een tetraëder, een kubus en een bilunabirotunda met bijbehorende netten.

- De onderlinge verbindingen tussen $(n-1)$-dimensionale elementen blijven behouden en traceerbaar;
- Topologische eigenschappen, zoals het aantal hoekpunten, ribben, facetten en hogere-dimensionale elementen, blijven constant.

Voorbeeld 3 (3-kubus)**.** Een kubus kan op verschillende manieren worden uitgevouwen tot een vlak patroon, waarvan de kruisvorm de meest bekende is:



Figuur 4.2: Één mogelijke netstructuur van een kubus, gestippelde lijnen geven doorgesneden verbindingen

Zoals eerder benoemd, blijven bij een ontwikkeling topologische eigenschappen, zoals het aantal hoekpunten, randen en vlakken constant. Echter lijkt dit in de bovenstaande illustratie niet het geval te zijn. Hoewel het aantal vlakken (6) gelijk

blijft, zijn er in dit geval 7 randen bijgekomen ($12 \rightarrow 19$) en het aantal hoekpunten is bijna verdubbeld ($8 \rightarrow 14$). Dit klopt, het aantal getekende randen en hoekpunten is daadwerkelijk veranderd, maar het aantal verschillende randen en hoekpunten is dat niet.

In de bovenstaande illustratie zijn er randen en hoekpunten dubbel getekend. Bij het 'opensnijden' van een kubus, om een net te construeren, worden er randen *in tweeën gesplitst*. In de netstructuur worden er sommige randen dus dubbel gerepresenteerd, maar de representaties verwijzen naar dezelfde rand, en dus blijft het aantal randen gelijk.

Omdat elke rand begrensd wordt door twee punten, verandert ook het aantal getekende punten tijdens het ontwikkelen.

4.2 Netstructuren in hogere dimensies

Netstructuren kunnen naar hogere dimensies worden uitgebreid. Zo wordt een 4-dimensionale kubus uitgevouwen naar acht 3-dimensionale kubbussen. Dit is vergelijkbaar met het uitvouwen van een 3-dimensionale kubus naar een netstructuur van zes vierkanten. De complexiteit neemt exponentieel toe met de dimensie, waardoor het steeds moeilijker wordt om ze intuïtief te begrijpen. Zo bestaat bijvoorbeeld een 5-dimensionale kubus uit tien 4-dimensionale kubussen of veertig 3-dimensionale kubussen.

4.3 Uitvouwregels en eigenschappen van netstructuren

Bij het creëren van netstructuren van n-dimensionale polytopen naar $(n-1)$-dimensionale representaties moeten de volgende fundamentele eigenschappen behouden blijven:

1. Alle $(n-1)$-dimensionale facetten van de originele polytoop moeten aanwezig zijn in de netstructuur;

2. De meetkundige eigenschappen van elk facet (zoals vorm en grootte) blijven behouden;

3. De connectiviteit tussen facetten moet bewaard blijven, zodat de originele polytoop gereconstrueerd kan worden;

4. De onderlinge hoeken tussen facetten in de netstructuur moeten zodanig zijn dat de facetten bij reconstructie correct aansluiten.

Deze eigenschappen zorgen ervoor dat de netstructuur een accurate representatie is van de originele polytoop en dat de topologische structuur behouden blijft.

4.4 Overlap in netstructuren

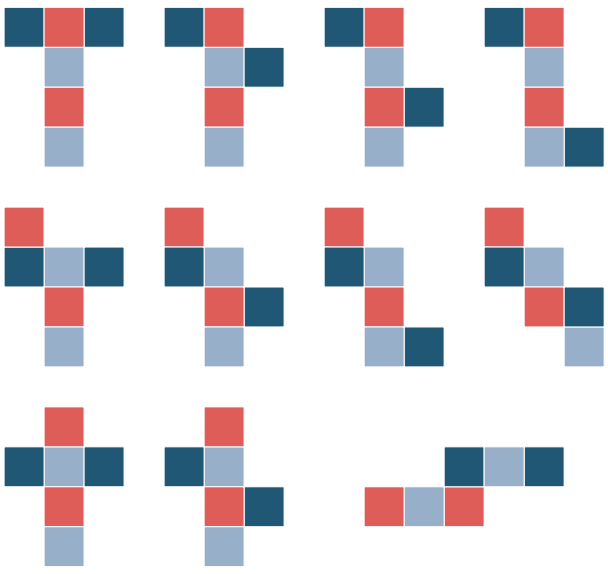
Een belangrijk aspect van netstructuren is dat ze zowel met als zonder overlap kunnen voorkomen. Dit leidt tot een fundamenteel onderscheid tussen verschillende types netstructuren.

4.4.1 Niet-overlappende netstructuren

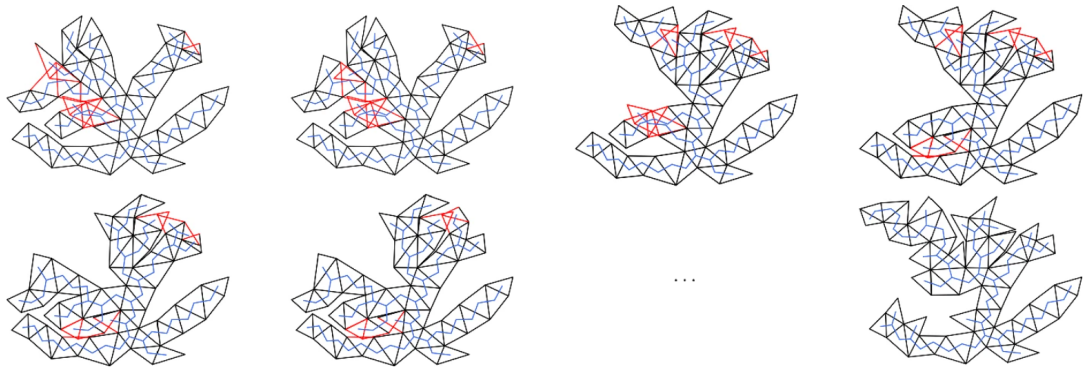
Deze worden vaak aangeduid als 'goede' of 'valide' netstructuren en hebben de volgende eigenschappen:

- De $(n-1)$-dimensionale facetten hebben geen onderlinge doorsnede in de netstructuur;
- Ze zijn isometrisch met de originele facetten van de n-dimensionale polytoop;
- Ze behouden alle topologische eigenschappen van de originele polytoop.

Voorbeeld 4. Voor de 3-kubus bestaan er precies elf verschillende (niet-overlappende) netstructuren. Voor hogere-dimensionale kubussen neemt dit aantal snel toe, voor meer informatie zie appendix C.2.



Figuur 4.3: Alle netstructuren van een 3-dimensionale kubus



Figuur 4.4: Overlappende delen met rood aangegeven (uit [43])

4.4.2 Overlappende netstructuren

Deze netstructuren zijn wiskundig gezien ook geldig, maar hebben andere eigenschappen:

- De $(n-1)$-dimensionale facetten kunnen elkaar doorsnijden in de netstructuur;
- Ze behouden nog steeds de isometrische eigenschappen van individuele facetten;
- Ze komen vaker voor naarmate de dimensie van de polytoop toeneemt.

De complexiteit van het vinden van niet-overlappende netstructuren neemt significant toe met elke extra dimensie. Voor sommige n-dimensionale polytopen is het zelfs een open vraag of er überhaupt niet-overlappende netstructuren bestaan.

4.4.3 Existentie van niet-overlappende netstructuren

Een fundamentele vraag in de studie van netstructuren is of er voor elke polytoop een niet-overlappende netstructuur bestaat. Dit is niet vanzelfsprekend en vormt een belangrijk onderzoeksgebied binnen de geometrie.

Voor sommige relatief eenvoudige polytopen, zoals de regelmatige platonische lichamen, is bekend dat er altijd niet-overlappende netstructuren bestaan. Echter, voor complexere polytopen in hogere dimensies is dit niet altijd gegarandeerd.

Stelling 4.4.1 (Stelling van Alexandrov)**.** *Elke convexe 3-dimensionale polytoop heeft ten minste één niet-overlappende netstructuur.*

Opmerking 3. De generalisatie van deze stelling naar hogere dimensies blijft een open vraag in de wiskunde.

Hoofdstuk 5

Grafentheorie

5.1 Inleiding

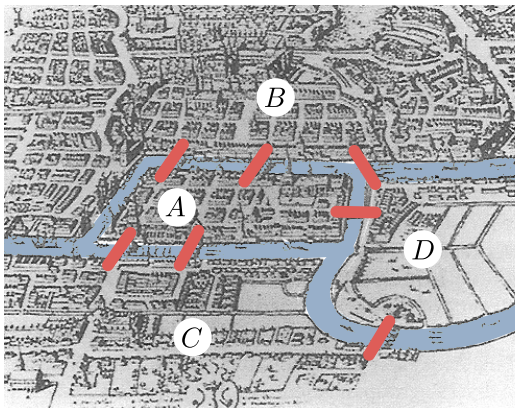
5.1.1 Wat is grafentheorie?

Grafentheorie is een tak van de wiskunde die zich bezighoudt met de studie van grafen, abstracte structuren die bestaan uit knopen (of punten) en de verbindingen (of randen) tussen deze knopen. Grafen bieden krachtige modellen om verschillende soorten relaties en netwerken te beschrijven, van sociale netwerken tot netwerken in computerwetenschappen en biologie. In de moderne wereld zijn grafen essentieel voor het begrijpen van complexe systemen en processen, zoals verkeersstromen, communicatiepatronen en gegevensstructuren.

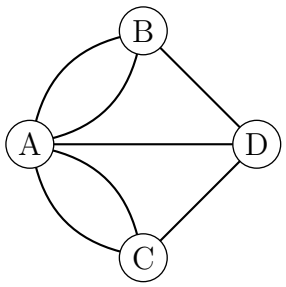
Dit hoofdstuk heeft als doel om de fundamenten van grafentheorie te verkennen, te beginnen met de basisconcepten en de verschillende soorten grafen. We zullen ook ingaan op de belangrijkste algoritmen die worden gebruikt bij het analyseren en oplossen van problemen binnen deze discipline, met de focus op de technieken die van belang zijn binnen de hogere-dimensionale meetkunde. Veel van de hier behandelde begrippen en theorieën zullen in onze methode toegepast worden.

5.1.2 Geschiedenis van grafentheorie

Historisch gezien vindt de oorsprong van grafentheorie zijn wortels in de 18e eeuw met het beroemde 'Seven Bridges of Königsberg' probleem, gepresenteerd door de wiskundige Leonhard Euler. Dit probleem stelde de vraag of het mogelijk was om bij een wandeling alle zeven bruggen in de stad Königsberg slechts één keer te oversteken, wat leidde tot de ontwikkeling van de basisprincipes van grafentheorie. Eulers werk markeert niet alleen een keerpunt in de wiskunde, maar heeft ook invloed gehad op andere disciplines zoals informatica, operationele research en zelfs kunstmatige intelligentie [29].



(a) Kaart van Königsberg



(b) Representatie als een graaf

Figuur 5.1: 'Seven Bridges of Königsberg'

5.2 Basisconcepten

5.2.1 Grafen

Definitie 3. Een graaf G is een paar $G = (V, E)$, waarbij:

- V een verzameling is van knopen (of vertexen),
- E een verzameling is van lijnen (of randen) die de knopen verbinden.

5.2.2 Gerichte en ongerichte grafen

Grafen kunnen worden onderverdeeld in gerichte en ongerichte grafen, afhankelijk van de aard van de verbindingen:

- **Ongerichte graaf**: In een ongerichte graaf hebben de randen geen richting. Een rand wordt weergegeven als een ongesorteerd paar $\{u, v\}$, waarbij u en v knopen zijn. Dit betekent dat de verbinding tussen u en v wederzijds is, en de volgorde van de knopen geen rol speelt.
- **Gerichte graaf**: In een gerichte graaf hebben de randen een richting. Een rand wordt weergegeven als een geordend paar (u, v), waarbij u het beginpunt is en v het eindpunt. Dit betekent dat de verbinding van u naar v niet noodzakelijk hetzelfde is als van v naar u.

5.2.3 Gewogen en ongewogen grafen

Grafen kunnen ook worden onderverdeeld in gewogen en ongewogen grafen, afhankelijk van de aanwezigheid van gewichten op de randen:

- **Ongewogen graaf**: In een ongewogen graaf zijn de randen slechts aanwezig om aan te geven dat er een verbinding bestaat tussen twee knopen. Er zijn geen aanvullende waarden (gewichten) aan de randen gekoppeld.

- **Gewogen graaf**: In een gewogen graaf heeft elke rand een gewicht, dat wordt weergegeven door een getal. Dit gewicht kan bijvoorbeeld de afstand, kosten of tijd voorstellen die nodig is om van de ene knoop naar de andere te gaan. Voor een rand (u, v) wordt het gewicht genoteerd als $w(u, v)$.

5.2.4 Knopen en randen

In grafentheorie worden de knopen (of vertexen) gezien als de basiselementen van een graaf. Deze knopen kunnen bijvoorbeeld steden, objecten of andere entiteiten vertegenwoordigen, afhankelijk van de toepassing. De randen (of lijnen) verbinden de knopen met elkaar en beschrijven de relaties of interacties tussen hun. In het geval van een gewogen graaf kunnen deze randen extra informatie bevatten, zoals een afstand of een kostenwaarde.

5.2.5 Graad van een knoop

De graad van een knoop in een graaf is het aantal randen dat met die knoop verbonden is. Afhankelijk van het type graaf kunnen we de graad als volgt definiëren:

- Voor een ongerichte graaf is de graad van een knoop simpelweg het aantal randen die aan die knoop verbonden zijn. Bijvoorbeeld, als knoop v_0 verbonden is met knopen $v_1, v_2, \ldots, v_k$, dan is de graad van v_0, genoteerd als $\deg(v_0)$, gelijk aan k.

- Voor een gerichte graaf splitsen we de graad op in:

  - **In-graad** ($\deg^-(v)$): Het aantal inkomende randen naar de knoop v. Dit is het aantal randen van andere knopen naar v.
  - **Uit-graad** ($\deg^+(v)$): Het aantal uitgaande randen vanuit de knoop v. Dit is het aantal randen van v naar andere knopen.

5.3 Grafen representaties

Hoewel grafen als abstract concept in veel gevallen voldoende werken, is het voor dit onderzoek belangrijk aandacht te besteden aan hoe grafen (digitaal) numeriek gerepresenteerd kunnen worden, zodat hier algoritmen op losgelaten kunnen worden. Er zijn verschillende manieren waarop dit kan, waarbij elke methode eigen plus- en

minpunten heeft. Zo is de een beter geschikt voor complexe grafen, terwijl met de andere makkelijker te werken is.

Er worden verschillende manieren besproken om grafen te begrijpen, buiten alleen hun visuele en concrete vorm.

5.3.1 Adjacency list

Een adjacency list is een datastructuur om een graaf te representeren door elke knoop te associëren met een lijst van naburige knopen. Deze methode is bijzonder geschikt voor dunbevolkte grafen, omdat alleen de aanwezige randen worden opgeslagen.

Neem de graaf uit figuur 5.1b. De adjacency list van deze graaf ziet er als volgt uit:

- $a \to \{b, c, d\}$
- $b \to \{a, c\}$
- $c \to \{a, b\}$
- $d \to \{a\}$

Voor- en nadelen

De adjacency list biedt efficiënte toegang tot de buren van een knoop. Het opvragen van alle buren van een knoop met graad d kost slechts $O(d)$ tijd, omdat alleen de naburige knopen worden doorlopen. Deze efficiëntie maakt de adjacency list geschikt voor grafen met veel knopen en relatief weinig randen.

Echter, het testen of er een rand bestaat tussen twee specifieke knopen (bijvoorbeeld tussen a en b) vereist het doorzoeken van de lijst van naburige knopen en kost daarom ook $O(d)$ in het slechtste geval, waarbij d de graad van de knoop is. Dit kan minder efficiënt zijn dan gebruik te maken van de adjacency matrix voor dichtbevolkte grafen.

Opmerking 4. De notatie $O(f(n))$ beschrijft de tijdcomplexiteit van een algoritme in termen van een functie $f(n)$, waarbij n een maat is voor de grootte van de invoer (bijvoorbeeld het aantal knopen of randen in een graaf). De functie $f(n)$ geeft aan hoe de benodigde tijd (of ruimte) toeneemt naarmate de grootte van de invoer groeit. Als bijvoorbeeld de tijdcomplexiteit $O(d)$ is, betekent dit dat de tijd lineair toeneemt met de graad d van een knoop. Als de tijdcomplexiteit $O(d^2)$ is, betekent dit dat de tijd kwadratisch toeneemt met de graad van een knoop, wat doorgaans een hogere kosten met zich meebrengt dan een lineaire relatie. Hoe steiler de functie (bijvoorbeeld bij hogere machten van d), hoe langzamer het algoritme zal draaien naarmate de invoer groter wordt.

5.3.2 Adjacency matrix

Een adjacency matrix is een datastructuur om een graaf te representeren als een 2-dimensionale matrix. Elke rij en kolom in de matrix vertegenwoordigt een knoop, en een element in de matrix geeft aan of er een rand bestaat tussen twee knopen. Dit maakt de adjacency matrix efficiënt voor dichtbevolkte grafen, omdat de aanwezigheid van elke mogelijke rand expliciet wordt weergegeven.

De adjacency matrix van de eerder behandelde graaf uit figuur 5.1b, ziet als volgt uit:

	a	b	c	d
a	0	1	1	1
b	1	0	1	0
c	1	1	0	0
d	1	0	0	0

In deze matrix vertegenwoordigt de waarde 1 een rand tussen twee knopen, terwijl 0 aangeeft dat er geen rand is. Zo zien we bijvoorbeeld dat knoop a is verbonden met knopen b, c, en d, wat overeenkomt met de structuur van de graaf.

Voor- en nadelen

Een adjacency matrix biedt efficiënte toegang om te testen of er een rand bestaat tussen twee knopen: deze operatie kost slechts $O(1)$ tijd, omdat elk element in de matrix direct kan worden opgezocht door de rijen en kolommen te indexeren. Dit maakt de adjacency matrix bijzonder geschikt voor toepassingen waarbij veelvuldig de aanwezigheid van randen moet worden gecontroleerd, zoals bij dichtbevolkte grafen.

Aan de andere kant kan de adjacency matrix minder ruimte-efficiënt zijn voor dunbevolkte (sparse) grafen, aangezien er ook geheugen wordt gereserveerd voor randen die niet bestaan. De ruimtecomplexiteit van de adjacency matrix is $O(\#V^2)$, waarbij $\#V$ het aantal knopen in de graaf is. Hierdoor kan de adjacency list in veel gevallen een geschiktere keuze zijn voor grote, dunbevolkte grafen.

5.3.3 Incidence matrix

Een incidence matrix is een datastructuur om een graaf te representeren als een matrix waarbij de rijen de knopen (vertices) vertegenwoordigen en de kolommen de randen (edges). Elk element in de matrix geeft aan of een knoop in verbinding staat met een bepaalde rand. Dit type representatie is vooral nuttig in toepassingen waarbij zowel knopen als randen belangrijke kenmerken bevatten of veelvuldig moeten

worden geanalyseerd.

Neem de graaf uit figuur 5.1b als voorbeeld. Deze graaf heeft de knopen a, b, c, en d en de randen e_1, e_2, e_3, en e_4, waarbij:

- e_1 verbindt a en b,
- e_2 verbindt a en c,
- e_3 verbindt a en d,
- e_4 verbindt b en c.

De incidence matrix voor deze graaf ziet er als volgt uit:

	e_1	e_2	e_3	e_4
a	1	1	1	0
b	1	0	0	1
c	0	1	0	1
d	0	0	1	0

Hier geeft 1 aan dat een knoop in verbinding staat met een rand en 0 dat er geen verbinding is. In dit voorbeeld is knoop a bijvoorbeeld verbonden met de randen e_1, e_2, en e_3, zoals weergegeven in de matrix.

Voor- en nadelen

De incidence matrix biedt een overzichtelijke representatie voor het analyseren van zowel knopen als randen, waardoor het mogelijk is snel te zien welke knopen bij welke randen betrokken zijn. Voor ongerichte grafen met $\#V$ knopen en $\#E$ randen heeft de incidence matrix een ruimtecomplexiteit van $O(\#V \cdot \#E)$.

Het testen van de aanwezigheid van een rand kost $O(1)$ tijd, net zoals bij de adjacency matrix. Echter, het doorlopen van alle buren van een knoop vereist dat alle kolommen in de betreffende rij worden gecontroleerd, wat $O(\#E)$ tijd kost. Dit maakt de incidence matrix minder efficiënt voor bewerkingen waarbij de nadruk ligt op de buren van een knoop, in vergelijking met de adjacency list en matrix.

De incidence matrix is dus vooral nuttig in situaties waarin het belangrijk is om randen als individuele objecten te behandelen, bijvoorbeeld bij toepassingen die eigenschappen per rand opslaan of in netwerkanalyses.

5.4 Spanning trees

Een spanning tree (opsommende boom) is een deelverzameling van een graaf die alle knopen bevat, maar enkel een minimale set van randen om een samenhangende

structuur te vormen zonder cykels. Met andere woorden, een spanning tree is een graaf die elke knoop verbindt met een andere knoop, zonder extra verbindingen die gesloten cykels creëren. Als een originele graaf G n knopen heeft, bevat elke spanning tree van G precies $n - 1$ randen.

Definitie 4. Voor een verbonden, ongerichte graaf $G = (V, E)$ is een spanning tree $T = (V, E_T)$ een deelgraaf waarbij:

- T alle knopen van G bevat,
- T een samenhangende structuur vormt zonder cykels,
- $\#E_T = \#V - 1$.

Een spanning tree is nuttig in toepassingen waar het belangrijk is om een netwerk te verbinden met zo min mogelijk verbindingen, zoals in computer-, telecommunicatie- en elektriciteitsnetwerken. In dergelijke netwerken kan een spanning tree helpen om de verbindingen te minimaliseren en zo de kosten en complexiteit te reduceren.

Voorbeeld 5. Neem de volgende graaf G met knopen a, b, c, en d, verbonden door randen zoals aangegeven in figuur 5.2. Een mogelijke spanning tree van G is een structuur waarin alle knopen verbonden zijn zonder dat er een cykel ontstaat.

(a) Een graaf (b) Voorbeeld van een spanning tree

Figuur 5.2: Een graaf en spanning tree

5.5 Minimal spanning trees

5.5.1 Wat is een minimal spanning tree?

Een minimal spanning tree (MST) is een specifieke soort spanning tree voor een gewogen graaf, waarbij het doel is om een spanning tree te vinden met het minimale gewichtsverschil over alle verbindingen. Dit betekent dat een MST niet alleen alle knopen van een graaf verbindt zonder cykels, maar ook de som van de gewichten van de gebruikte randen minimaliseert.

Definitie 5. Voor een verbonden, gewogen, ongerichte graaf $G = (V, E)$ met gewichten $w : E \to \mathbb{R}^+$ op de randen, is een minimal spanning tree een spanning tree $T = (V, E_T)$ die:

- alle knopen van G verbindt zonder cykels,
- de som van de gewichten van E_T minimaliseert:

$$\sum_{e \in E_T} w(e) \quad \text{is minimaal.}$$

5.5.2 Algoritmen voor minimal spanning trees

Twee bekende algoritmen voor het vinden van een MST zijn Kruskals algoritme en Prims algoritme. Beide algoritmen werken door het iteratief selecteren van de lichtste verbindingen, met enkele specifieke verschillen:

- **Kruskals algoritme**: Dit algoritme werkt door telkens de lichtste rand te kiezen die geen cykel creëert in de bouwende boom. Het is efficiënt voor grafen met gescheiden componenten en kan goed worden geïmplementeerd met behulp van gesorteerde lijsten en de union-find-datastructuur.
- **Prims algoritme**: Dit algoritme begint bij een willekeurige knoop en voegt telkens de lichtste rand toe die een nog niet-verbonden knoop verbindt aan de groeiende boom. Het is bijzonder efficiënt voor dichtbevolkte grafen wanneer het wordt geïmplementeerd met behulp van een prioriteitswachtrij (priority queue).

5.5.3 Toepassingen van minimal spanning trees

Minimal spanning trees worden vaak toegepast in netwerkanalyse om de kosten van het bouwen van netwerken te minimaliseren. Dit is van belang in de ontwerpkeuzes voor diverse netwerkstructuren, zoals:

- **Telecommunicatienetwerken**: Minimaliseert de kosten voor het leggen van kabels in een netwerk, terwijl elke locatie bereikt wordt.
- **Wegen- en spoorwegennetwerken**: Ontwerpt wegen of railsystemen die elke stad of halte met elkaar verbinden met zo min mogelijk kosten.
- **Computer- en datanetwerken**: Kiest kosteneffectieve verbindingen in datacenters of bij cloudserviceproviders.

Minimal spanning trees bieden zo een krachtig model om problemen op te lossen waarbij minimale verbindingen een sleutelrol spelen bij het optimaliseren van netwerkkosten.

Hoofdstuk 6

Casestudy deel 1: De n-kubus

In deze casestudy bekijken we de n-kubus, de generalisatie van de kubus in meerdere dimensies. De casestudy is in twee delen verdeeld, waarbij in het eerste deel de focus ligt op het definiëren en het bekijken van eigenschappen van de n-kubus. Op deze manier proberen we context te bieden bij dit theoretisch kader en een beter begrip te krijgen van hoe polytopen zich gedragen in verschillende dimensies.

6.1 Recursieve definitie van de n-kubus

6.1.1 Voorbeelden van de definitie

Hoewel iedereen bekend is met een 3D kubus, is het het beste om de definitie van de n-kubus vanaf het begin op te bouwen; dit begin is een nuldimensionale vertex.

0 dimensies Een vertex vormt, zoals eerder aangetoond, de basis voor elke polytoop; met een verzameling vertexen is elke polytoop te definiëren. Maar wat maakt deze vertexen nu precies zo speciaal? Een vertex kan bestaan in nul dimensies. Dit komt doordat deze geen omvang, lengte, breedte, of hoogte heeft. Hiermee is een vertex het beginsel voor de n-kubus: de 0-kubus, het is immers het enige 0-dimensionale 'object'.

Een n-kubus kan op een recursieve manier gedefinieerd worden; door het modificeren van de $(n-1)$-kubus ontstaat de n-kubus. Deze stappen kunnen tot in het oneindige worden herhaald, waardoor deze methode een definitie van elke n-kubus biedt, met $n \in \mathbb{N}$.

Figuur 6.1: Een grafische weergave van de 0-kubus

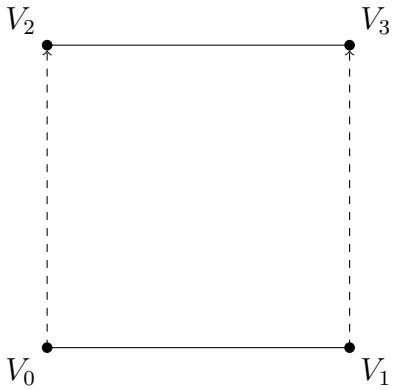
Omdat de basis van elke polytoop vertexen is, hebben wij deze vertexen genummerd met V_i, waarbij i een geheel getal is dat begint bij 0 en verder oploopt tot oneindig. Voordat we de recursieve stap een definitie geven, bekijken we eerst een aantal voorbeelden.

1 dimensie Een lijnsegment is ook wel te zien als een 1-dimensionale kubus. Dit lijnsegment kan uit de 0-kubus worden verkregen door deze te dupliceren en de twee vertexen met elkaar te verbinden. De originele 0-dimensionale kubussen begrenzen het 1-dimensionale lijnsegment. Elk punt op dit lijnsegment kan nu met een 1-tuple worden aangegeven. Dit kan gezien worden als de eerste stap in de recursieve definitie van de n-kubus.

Figuur 6.2: Een grafische weergave van de 1-kubus

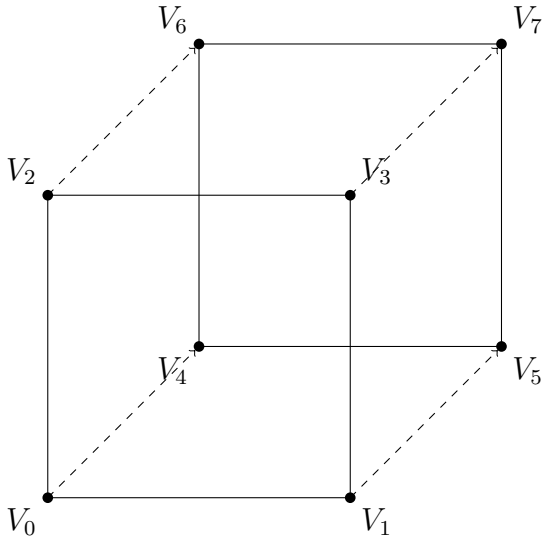
2 dimensies Dupliceer dit bovenstaande lijnsegment, verplaats deze in de richting loodrecht op het bestaande object en verbind elke vertex in de originele 1-kubus met de corresponderende vertex in het duplicaat. Het verkregen object bestaat uit $2\times$ het aantal punten van het vorige object, dit zijn er vier en daarmee is dit een vierhoek. Deze 2-dimensionale ruimte wordt begrensd door de vier lijnsegmenten.

Het aantal vertexen in de 2-kubus geven we aan met $\#V(2) = 4$ en het aantal 2D lijnsegmenten of randen (Engels: edges) aan met $\#E(2) = 4$.



Figuur 6.3: Een grafische weergave van de 2-kubus

3 dimensies Om de $(n+1)$-kubus met $n+1=3$ te verkrijgen, dupliceren we de n-kubus, verplaatsen we deze in de richting loodrecht op de n-kubus en verbinden we reeds elke vertex met haar duplicaat. De ruimte die door de vierkanten wordt begrensd is de 3-kubus.



Figuur 6.4: Een grafische weergave van de 3-kubus

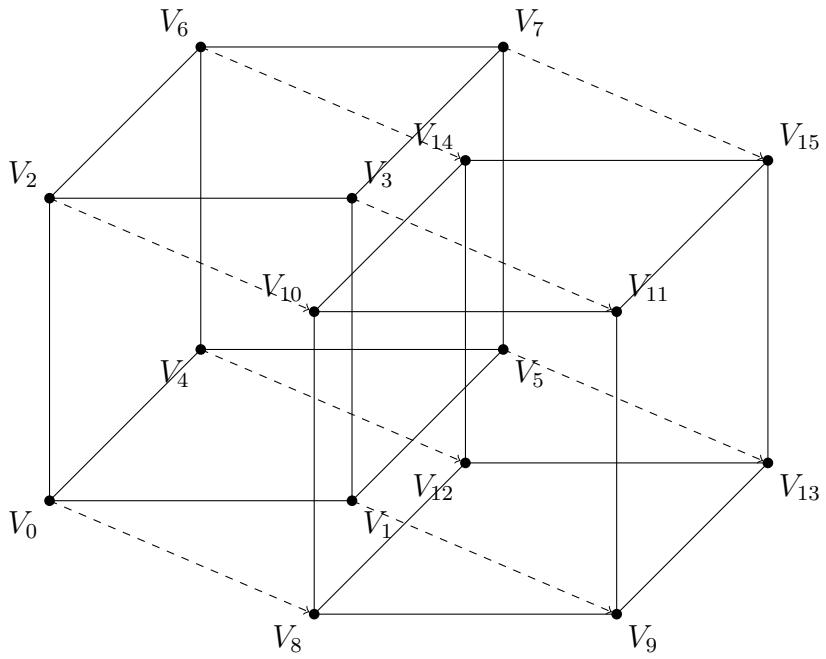
Het is nu mogelijk om het aantal randen en vertexen van de 3-kubus te berekenen. Omdat de vertexen worden verkregen door de 2-kubus te dupliceren, dupliceert ook het aantal vertexen ten opzichte van de 2-kubus.

$$\#V(3) = 2 \cdot \#V(2)$$
$$\#V(3) = 2 \cdot 4 = 8$$

Omdat de 2-kubus wordt gedupliceerd, verdubbelt ook het aantal randen. Echter worden er ook nog een aantal randen bijgetekend, namelijk één tussen elk hoekpuntenpaar. De formule voor het aantal randen van de 3-kubus is daarom:

$$\#E(3) = 2 \cdot \#E(2) + \#V(2)$$
$$\#E(3) = 2 \cdot 4 + 4 = 12$$

4 dimensies Dit patroon houdt stand bij de vierde dimensie en elke dimensie die daarop volgt. De 3-kubus wordt gedupliceerd, verplaatst over de 4e as (die voor ons niet te visualiseren is) en elk hoekpunt wordt verbonden met het nieuwe hoekpunt. De verkregen polytoop, die wordt begrensd door acht 3-kubussen, wordt naast de 4-kubus ook wel de hypercube of tesseract genoemd.



Figuur 6.5: Een poging tot grafische weergave van de 4-kubus

Het is belangrijk om te begrijpen dat de nieuwe kubus niet 'schuin' op de originele kubus is verschoven. De nieuwe kubus is in een nieuwe dimensie ($n = 4$) verschoven, maar hier kunnen wij ons geen voorstelling bij maken. Afbeelding 6.5 is dus meer een schematische weergave; alle onderdelen komen er in voor maar de verhoudingen kloppen niet. Omdat dit papier met een 2-dimensionaal oppervlak is, geldt dat overigens ook voor afbeelding 6.4.

De eerder genoemde formules voor het berekenen van het aantal vertexen en edges van de 3-kubus, zijn ook toepasbaar op de 4-kubus:

$$\#V(4) = 2 \cdot \#V(3)$$
$$\#V(4) = 2 \cdot 8 = 16$$

$$\#E(4) = 2 \cdot \#E(3) + \#V(3)$$
$$\#E(4) = 2 \cdot 12 + 8 = 32$$

6.1.2 Concrete definitie

De stappen die tot nu elke keer worden genomen, kunnen tot in het oneindige worden herhaald. Hiermee vormt dit een recursieve definitie van de n-kubus.

Definitie 6. Een n-dimensionale kubus, of n-kubus, is een polytoop die recursief kan worden gedefinieerd op de volgende manier:

- Voor $n = 0$ is de n-kubus een enkel punt (vertex).

- Voor $n \geq 0$ kan de $(n+1)$-kubus worden geconstrueerd door:

  1. De n-kubus te dupliceren;
  2. De gedupliceerde n-kubus over een vaste afstand langs de $(n+1)$-de as, orthogonaal aan de n dimensies van de oorspronkelijke kubus, te verplaatsen;
  3. Elk hoekpunt van de oorspronkelijke n-kubus te verbinden met het corresponderende hoekpunt van de gedupliceerde n-kubus met een lijnsegment.

Op deze manier wordt de $(n+1)$-kubus verkregen door de n-kubus uit te breiden met een extra dimensie.

6.2 n-kubus als oplossing van lineaire ongelijkheden

In aanvulling op de recursieve definitie van de n-kubus, kan deze ook worden opgevat als een verzameling van punten in de euclidische ruimte $\mathbb{R}^n$ die voldoen aan een systeem van lineaire ongelijkheden. Deze aanpak is nuttig voor het beschrijven van de geometrische grenzen van de n-kubus vanuit algebraïsch perspectief.

In wiskundige termen is de n-kubus het verzameldeel van alle punten $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ in $\mathbb{R}^n$ waarvoor de coördinaten voldoen aan de volgende ongelijkheden:

$$-1 \leq x_i \leq 1, \quad \text{voor alle } i = 1, 2, \ldots, n \tag{6.1}$$

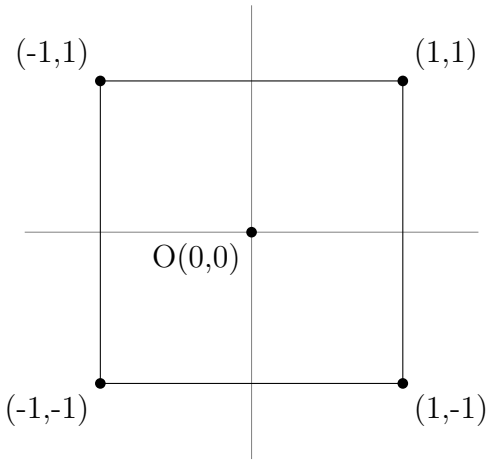
Dit betekent dat elk coördinaat van een punt in de n-kubus tussen -1 en 1 ligt.

6.2.1 Voorbeeld: 2-kubus (vierkant)

Een eenvoudig voorbeeld is de 2-kubus, ook wel bekend als een vierkant in de 2-dimensionale ruimte. Hierbij geldt voor elke vertex $x = (x_1, x_2)$:

$$-1 \leq x_1 \leq 1 \quad \text{en} \quad -1 \leq x_2 \leq 1$$

De verzameling van alle oplossingen voor dit systeem van ongelijkheden vormt een vierkant in het vlak $\mathbb{R}^2$, met de vier hoekpunten $(-1,-1)$, $(-1,1)$, $(1,-1)$, $(1,1)$. Dit vierkant is de 2-kubus, de hoekpunten vormen de uiterste waardes, maar elke oplossing van de ongelijkheden valt binnen de tweedimensionale regio omsloten door het vierkant.



6.2.2 Algemeen geval: n-kubus

De ongelijkheden voor de n-kubus kunnen ook worden geschreven in matrixvorm als een systeem van lineaire ongelijkheden:

$$A\boldsymbol{x} \leq \boldsymbol{b}$$

waarbij A een matrix is die de coëfficiënten van de ongelijkheden bevat, $\boldsymbol{x}$ de coördinatenvector is, en $\boldsymbol{b}$ een vector is die de rechterleden van de ongelijkheden bevat. Voor de n-kubus is deze matrix A eenvoudig omdat de grenzen voor elke coördinaat tussen -1 en 1 liggen. De ongelijkheden kunnen worden geschreven als:

$$\begin{pmatrix} 1 & 0 & \cdots & 0 \\ -1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ 0 & -1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \\ 0 & 0 & \cdots & -1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \leq \begin{pmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix} \tag{6.2}$$

Hieruit blijkt dat de n-kubus algebraïsch wordt omschreven als een systeem van $2n$ lineaire ongelijkheden die elk coördinaat van een punt beperken tussen -1 en 1.

6.3 De elementen van de n-kubus

Door de eerder beschreven recursieve definitie van de n-kubus 6 te bestuderen, zijn er algemene formules op te stellen voor de aantallen van verschillende elementen binnen een n-kubus.

Aantal vertexen

Tijdens de recursieve stap wordt telkens de $(n-1)$-kubus gedupliceerd om de n-kubus te verkrijgen. Hierdoor zal telkens het aantal vertexen verdubbelen. De recursieve basis bij $n = 0$ heeft 1 vertex. De formule voor het aantal vertexen V in een n-kubus luidt dus:

$$\#V(n) = 2 \cdot \#V(n-1) \tag{6.3}$$

$$\#V(n) = 2^n \tag{6.4}$$

Aantal randen

Tijdens het dupliceren van de $(n-1)$-kubus in de recursieve stap worden ook alle al bestaande randen gedupliceerd. Echter worden er daarna nieuwe randen bijgetekend, die ook in de formule meegenomen moeten worden. Omdat er een nieuwe rand ontstaat vanaf elke originele vertex naar de corresponderende nieuwe vertex, komt er elke keer het aantal vertexen in de $(n-1)$-kubus bij. De formule voor het aantal randen E is dus:

$$\#E(n) = 2 \cdot \#E(n-1) + \#V(n-1) \tag{6.5}$$

$$\#E(n) = 2 \cdot \#E(n-1) + n^{n-1} = n2^{n-1} \tag{6.6}$$

Algemene formule

We willen een algemene formule opstellen voor het aantal k-dimensionale elementen binnen een n-dimensionale hyperkubus. Hoewel inductie een logische keuze is vanwege de recursieve structuur van de n-kubus, kan het bewijs ook combinatorisch gegeven worden.

Stelling 6.3.1. *Voor een n-dimensionale hyperkubus geldt de volgende formule voor het aantal k-dimensionale elementen $\#F_k(P_n)$:*

$$\#F_k(P_n) = \binom{n}{k} \cdot 2^{n-k} \tag{6.7}$$

waarbij $0 \leq k \leq n$.

Bewijs. We bewijzen deze formule door te redeneren over de manier waarop k-dimensionale elementen binnen een n-dimensionale hyperkubus worden gevormd.

Selectie van de variabele dimensies

Om een k-dimensionaal element in een n-dimensionale ruimte te vormen, kiezen we k coördinaatrichtingen die vrij zijn en kunnen variëren tussen 0 en 1. De overige $n-k$ coördinaatrichtingen worden vastgezet op een vaste waarde (0 of 1).

Het aantal manieren om deze k variabele coördinaatrichtingen te selecteren uit de n beschikbare richtingen is $\binom{n}{k}$. De notatie $\binom{n}{k}$ geeft het aantal manieren weer om k elementen te kiezen uit n, en wordt berekend als:

$$\binom{n}{k} = \frac{n!}{k!(n-k)!} \tag{6.8}$$

.

Vastzetten van de vaste coördinaatrichtingen

Voor de $n-k$ coördinaatrichtingen die vaststaan, kunnen we elke coördinaat onafhankelijk op twee mogelijke waarden zetten (in dit geval -1 of 1). Omdat er $n-k$ vaste coördinaatrichtingen zijn, zijn er in totaal 2^{n-k} mogelijke combinaties om deze vast te zetten.

Combineren van de keuzes

Voor elke keuze van de k variabele coördinaatrichtingen zijn er 2^{n-k} manieren om de resterende coördinaatrichtingen vast te zetten. Het totale aantal k-dimensionale elementen wordt dus gegeven door:

$$\#F_k(P_n) = \binom{n}{k} \cdot 2^{n-k}.$$

Deze combinatorische redenering leidt ons tot de gewenste formule. □

6.4 De coördinaten van de hoekpunten in een n-kubus

Beschouw een n-dimensionale kubus waarbij de lengte van elke rand gelijk is aan 1 en een van de hoekpunten zich bevind in de oorsprong. Elk hoekpunt van de n-kubus, aangeduid als V_i, bevindt zich op een afstand van 1 van de aangrenzende hoekpunten waarmee het verbonden is. Hierbij geldt dat i varieert van 0 tot en met 2^n-1, oftewel: $0 \leq i \leq 2^n-1$, waarbij i een geheel getal is.

We nummeren de hoekpunten van een n-kubus van 0 tot en met 2^n-1, waarbij i een geheel getal is en $0 \leq i \leq 2^n-1$. Om de coördinaten van een hoekpunt V_i te

bepalen in de n-dimensionale ruimte, kunnen we het volgende doen:

De coördinaten van elk hoekpunt V_i kunnen worden gevonden door het getal i om te zetten naar een binaire representatie. Elk van de n cijfers in deze binaire representatie komt overeen met een coördinaat van V_i. Als het k-de binaire cijfer van i gelijk is aan 0, dan is de k-de coördinaat van V_i gelijk aan 0; als het binaire cijfer 1 is, dan is de k-de coördinaat van V_i gelijk aan 1.

Op deze manier kunnen we de coördinaten van elk van de 2^n hoekpunten bepalen.

Stelling 6.4.1. *De coördinaten van de hoekpunten van een n-dimensionale kubus met een zijde van lengte 1 en een hoekpunt in de oorsprong, worden gegeven door de binaire representaties van hun index i ($0 \leq i \leq 2^n - 1$). Elk van de n bits in deze binaire representatie bepaalt de waarde (0 of 1) van de coördinaat in de corresponderende dimensie.*

Bewijs. We willen bewijzen dat de coördinaten van elk hoekpunt van een n-kubus kunnen worden beschreven door een binaire representatie van het indexgetal i.

Beschouw een n-kubus als het Cartesische product van n intervallen $[0, 1]$. Elk hoekpunt van de n-kubus komt overeen met een unieke n-tuple $(x_1, x_2, \ldots, x_n)$, waarbij elke x_i óf 0 óf 1 is. Dit komt doordat elk coördinaat van een hoekpunt in een n-dimensionale kubus alleen de waarden 0 of 1 kan aannemen, omdat de kubus langs elke dimensie een lengte van 1 heeft.

Nu nummeren we de hoekpunten door middel van het getal i, waarbij $0 \leq i \leq 2^n - 1$. De binaire representatie van i bestaat uit n bits, wat overeenkomt met de n dimensies van de kubus. Elk bit in de binaire representatie bepaalt of de coördinaat van het corresponderende hoekpunt in die dimensie 0 of 1 is.

Dus, als i bijvoorbeeld 5 is in een 3-kubus, is de binaire representatie van 5 gelijk aan 101_2 (de subscript 2 geeft aan dat dit een getal in het binaire talstelsel is), wat betekent dat de coördinaten van V_5 $(1, 0, 1)$ zijn.

Aangezien er 2^n mogelijke binaire reeksen van lengte n zijn, zijn er precies 2^n hoekpunten in de n-kubus, elk overeenkomend met een unieke binaire representatie.

□

Conclusie De coördinaten van de hoekpunten van de eerder omschreven n-kubus worden bepaald door de binaire representatie van hun indexgetal i, waarbij elk cijfer in de binaire representatie de waarde 0 of 1 van het coördinaat in die dimensie weergeeft.

Deel II

De methode

Hoofdstuk 7

Inleiding tot de methode

In dit deel behandelen we onze methode voor het ontwikkelen van n-dimensionale polytopen. Deze is ontworpen op basis van de eigenschappen die elke polytoop heeft, ongeacht het aantal dimensies. Het grootste gedeelte van de benodigde eigenschappen, zijn in het theoretisch kader behandeld.

Onze methode is geïnspireerd op de methode ontwikkeld door Straub en Prautzsch in 2011 [39] voor het ontwikkelen van 3D-modellen naar papieren 2D-bouwpaketten. Wij hebben deze principes in een grotere context toegepast en uitgebreid naar de vierde dimensie en alles wat daar boven komt.

7.1 Is er een net voor elke polytoop?

Voordat we de methode in detail omschrijven, is het belangrijk om vast te stellen of polytopen überhaupt uit te vouwen zijn en voor welke polytopen het algoritme geen uitkomst hoeft te hebben, omdat er voor deze polytopen ook geen net bestaat.

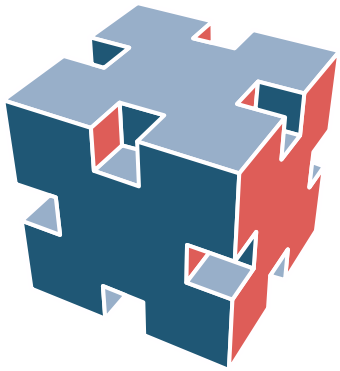
7.1.1 Netten zonder overlap

Stelling 7.1.1. *Er is niet voor elke polytoop een net te vinden, waarbij dit net geen overlap vertoont met zichzelf [13] [14].*

Bewijs. De stelling wordt bewezen door tegenvoorbeeld. In [13] wordt aangetoond dat er specifieke klassen van veelvlakken bestaan waarvoor elk mogelijk net zichzelf overlapt. Een voorbeeld hiervan, origineel omschreven in [14], is afgebeeld in fig. 7.1. □

7.1.2 Netten met overlap

Stelling 7.1.2. *Elke polytoop kan worden uitgevouwen naar een lagere dimensie als overlap toegestaan is.*



Figuur 7.1: Polyhedra zonder niet-overlappende netstructuur (op basis van [13])

Bewijs. Een polytoop in n-dimensionale ruimte bestaat uit $(n-1)$-dimensionale facetten, die elk in een $(n-1)$-dimensionale affiene subruimte liggen. Door de facetten los te snijden, kunnen ze individueel in de $(n-1)$-dimensionale ruimte worden geplaatst zonder beperkingen op hun onderlinge positionering.

Aangezien overlap is toegestaan, kunnen alle facetten vrij naast of over elkaar worden geprojecteerd. Elk $(n-1)$-dimensionaal facet past per definitie volledig in deze ruimte, waardoor het representatief blijft voor de originele structuur. Een expliciete constructie verloopt als volgt:

1. Kies een facet en positioneer het in de $(n-1)$-dimensionale ruimte.
2. Snijd de overige facetten los en plaats deze eveneens in de $(n-1)$-dimensionale ruimte, zonder restricties op overlap.
3. Herhaal totdat alle facetten zijn geprojecteerd.

Omdat overlap geen beperking is, bestaat er altijd een configuratie waarin alle facetten worden opgenomen. Dit bewijst dat elke polytoop kan worden uitgevouwen naar een lagere dimensie als overlap is toegestaan. □

7.2 Opbouw van de methode

Onze methode voor het ontwikkelen van polytopen bestaat uit een aantal stappen. Deze stappen definiëren de polytoop, delen deze op in kleinere stukken (facetten), en verplaatsen deze naar en binnen een lagere dimensie.

1. Representatie van polytoop als een graaf;
2. Vinden van de optimale spanning tree;
3. Van spanning tree naar netstructuur.

Hoofdstuk 8

Representatie polytoop als een graaf

Binnen het vraagstuk van het ontwikkelen van polytopen, is de exacte geometrische structuur irrelevant. Wel is belangrijk hoe elementen van deze structuur zich onderling verhouden. Het is dus niet nodig om binnen het algoritme te rekenen en redeneren met de exacte polytoop, zo lang er maar een manier is om de benodigde informatie op een andere manier te behouden.

Dit doel is te bereiken door middel van een graaf. Grafen zijn zeer geschikt voor het aangeven van onderlinge relaties, zoals in dit geval de relaties tussen elementen van de polytoop.

Voorbeeld 6. De twee objecten uit fig. 8.1 zijn verschillend. De geometrische structuur is gelijk, maar de (onderlinge) ligging van de hoekpunten verschilt. Bij het ontwikkelen van deze objecten maakt dit alleen niets uit; elk vlak is nog steeds verbonden aan dezelfde andere vlakken. Ook blijft het aantal elementen tussen de objecten gelijk.

(a) Een kubus (b) Vergelijkbaar object

Figuur 8.1: Objecten met vergelijkbare eigenschappen

Zoals besproken in hoofdstuk 3 bestaat elke n-dimensionale polytoop uit k-dimensionale elementen, met $k < n$. Bij het uitvouwen van polytopen zijn vooral

de facetten ($k = n - 1$) van belang. De facetten kunnen verbonden zijn, of niet verbonden zijn. Wanneer dit het geval is, wordt later omschreven.

Een graaf kan op meerdere manieren gemaakt worden op basis van een polytoop. In dit geval stellen we als volgt een graaf op die de onderlinge verbindingen tussen facetten weergeeft:

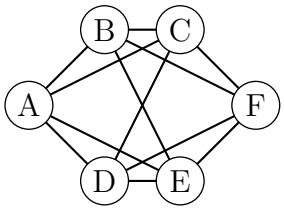
Definitie 7. Een **facet-adjacentiegraaf** is een graaf die de structuur van een polytoop weergeeft door middel van de facetten en hun onderlinge relaties. In deze graaf:

1. **Knooppunten** vertegenwoordigen de facetten van de polytoop. Bij een n-dimensionale polytoop zijn dit de $(n-1)$-dimensionale facetten.

2. **Randen** worden geplaatst tussen knooppunten als de bijbehorende facetten aan elkaar grenzen.

Een facet-adjacentiegraaf toont de **adjacentiën** tussen de facetten van een polytoop, zonder verdere details over de kleinere dimensies.

Voorbeeld 7. Voor beide objecten die omschreven zijn in voorbeeld 6 is facet-adjacentiegraaf identiek, omdat ze volgens de definitie identieke relaties tussen hun elementen hebben. De graaf ziet er (voor beide objecten) als volgt uit:



Figuur 8.2: Graaf van een object met 3-kubus-achtige structuur

In deze graaf staat elke knoop dus voor een facet, in dit geval een 2-dimensionaal vlak. Als twee vlakken aan elkaar grenzen, zijn de knopen met elkaar verbonden. In de grafische weergave van de graaf is te zien hoe elke knoop een graad van 4 heeft. Dit komt overeen met hoe elk vlak in het object met 4 andere vlakken verbonden is, wat weer komt doordat elk vlak een 'vierkant' (lees vlak met vier randen) is.

8.1 Recursief: vooruit

Dit principe werkt recursief. Elke facet is immers te zien als een nieuwe, $(n-1)$-dimensionale, polytoop. Deze bestaat weer uit $(n-2)$-dimensionale 'facetten'. (Het is gevaarlijk hier over facetten te spreken, aangezien er geen $(n-1)$ geldt, maar we kijken relatief.)

In een facet-adjacentiegraaf geeft elke knoop een polytoop van een dimensie lager weer. Deze polytopen van een lagere dimensie, hebben elk hun eigen (unieke) graaf. De recursie eindigt met grafen die lijnstukken weergeven (de knopen geven dan vertexen weer).

Deze benadering resulteert niet voor elke polytoop in een unieke graaf. De twee objecten uit voorbeeld 6 resulteren nog steeds in identieke grafen met deze methode. Voor het ontwikkelen van een polytoop maakt dit niet uit, maar om een polytoop volledig te definiëren moet er een extra stap gebeuren: Elk vertex moet een coördinaat krijgen. Als zowel de posities van alle hoekpunten in de n-ruimte, als de onderliggende grafen zijn vastgesteld, is de gehele n-dimensionale polytoop gedefiniëerd.

8.2 Recursief: achteruit

Hoewel het algoritme zoals net omschreven aan het begin logisch lijkt, wordt het al snel ingewikkeld als je het proces een paar stappen volgt. Om deze principes inzichtelijker te maken werkt het beter om het algoritme achteruit te volgen; beginnend bij de 0-dimensionale hoekpunten.

Stap 0

Zet elk hoekpunt van de polytoop in een verzameling F_0 van alle hoekpunten. Elk element $F_{0,i}$ in deze verzameling stelt een uniek hoekpunt van de polytoop voor, waarbij i de index van het hoekpunt is. Formeel noteren we:

$$F_0 = V = \{F_{0,0}, F_{0,1}, F_{0,2}, \dots\} \tag{8.1}$$

Stap 1

Maak een verzameling F_1 van alle 1-dimensionale elementen (of randen) van de polytoop. Elk element in F_1 is een lijnstuk dat precies twee hoekpunten uit F_0 verbindt. Dit kan formeel worden weergegeven als:

$$F_1 = E = \{F_{1,0}, F_{1,1}, F_{1,2}, \dots\} \tag{8.2}$$

Hierbij staat elk $F_{1,i}$ specifiek voor een lijnstuk dat een paar hoekpunten uit F_0 verbindt. Elke rand in F_1 vormt op zichzelf een graaf met twee knopen (de hoekpunten) en één rand. In de context van facetten wordt deze graaf gezien als een facet-adjacentiegraaf voor dat specifieke lijnstuk, waar het lijnstuk een 1-dimensionale polytoop is en de hoekpunten de facetten. Elk lijnstuk $F_{1,i}$ laat dus zien hoe twee hoekpuntent $F_{0,j}, F_{0,k}$ met elkaar verbonden zijn.

Stap 2

Maak vervolgens een verzameling F_2 van alle 2-dimensionale elementen van de polytoop. Elk element in F_2 stelt een vlak voor dat gevormd wordt door een verzameling randen uit F_1 die een gesloten contour omsluiten. We kunnen deze verzameling noteren als:

$$F_2 = \{F_{2,0}, F_{2,1}, F_{2,2}, \dots\} \tag{8.3}$$

Hierbij vertegenwoordigt elk $F_{2,i}$ een specifiek vlak dat randen uit F_1 bevat en aan elkaar grenst volgens een bepaalde structuur. Elk vlak in F_2 kan ook weer worden opgevat als een facet-adjacentiegraaf die de relaties tussen de randen in dat facet weergeeft. Binnen deze graaf vormen de randen uit F_1, bijvoorbeeld $F_{1,i}, F_{1,j}, F_{1,k}, \dots$, de knooppunten. Deze knooppunten zijn telkens met elkaar verbonden als de randen elkaar in een hoekpunt uit F_0 raken.

Stap 3

Maak een verzameling F_3 van alle 3-dimensionale elementen van de polytoop. Elk element in F_3 stelt een 3-dimensionaal element voor die wordt gevormd door een verzameling vlakken uit F_2 die gezamenlijk een gesloten oppervlak omsluiten. Formeel kunnen we deze verzameling reeds als volgt noteren:

$$F_3 = \{F_{3,0}, F_{3,1}, F_{3,2}, \dots\} \tag{8.4}$$

Hierbij vertegenwoordigt elk $F_{3,i}$ een specifiek 3-dimensionaal volume dat vlakken uit F_2 bevat en aan elkaar grenst volgens een bepaalde structuur. Elk volume in F_3 kan weer worden opgevat als een facet-adjacentiegraaf.

$$\vdots$$

Stap $(n-1)$

Maak een verzameling F_{n-1} van alle $(n-1)$-dimensionale elementen van de polytoop. Elk element in F_{n-1} is een facet van de polytoop en wordt gevormd door een verzameling $(n-2)$-dimensionale elementen uit F_{n-2}, die gezamenlijk een gesloten grens vormen. Formeel kunnen we deze verzameling als volgt noteren:

$$F_{n-1} = \{F_{n-1,0}, F_{n-1,1}, F_{n-1,2}, \dots\} \tag{8.5}$$

Hierbij stelt elk $F_{n-1,i}$ een specifiek facet van de polytoop voor, bestaande uit $(n-2)$-dimensionale elementen die aan elkaar grenzen volgens een bepaalde structuur. Elk

facet in F_{n-1} kan worden opgevat als een facet-adjacentiegraaf waarin de knooppunten de $(n-2)$-dimensionale elementen zijn, en de verbindingen bestaan als deze elementen een gemeenschappelijke $(n-3)$-dimensionale rand delen.

Stap n

Maak een verzameling F_n die bestaat uit het enige n-dimensionale element van de polytoop: de polytoop zelf. Dit is het volledige object dat wordt begrensd door alle $(n-1)$-dimensionale facetten uit F_{n-1}. Formeel noteren we:

$$F_n = \{F_{n,0}\} \tag{8.6}$$

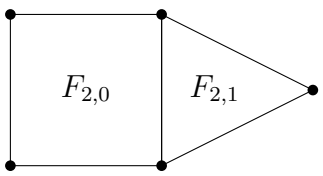
waarbij $F_{n,0}$ specifiek staat voor de originele n-dimensionale polytoop. Dit is het enige element in F_n, en het bevat alle onderliggende hiërarchische structuren van hoekpunten (F_0), randen (F_1), vlakken (F_2), enzovoort, tot aan de $(n-1)$-dimensionale facetten.

8.3 Wanneer grenzen facetten aan elkaar?

Bij het opstellen van de bovenstaande grafen, staat centraal dat er randen getekend worden tussen knopen, als de facetten elkaar raken. De vraag die daaruit volgt, is: wanneer grenzen facetten aan elkaar?

Om hier achter te komen, kan je naar de graaf van elke facet kijken. Neem twee elementen waarbij de dimensionaliteit van de affiene-deelruimte gelijk is: $F_{d,i}, F_{d,j}$. Bevatten de facet-adjacentiegrafen van deze elementen hetzelfde element uit F_{d-1}, dan grenzen de facetten aan elkaar.

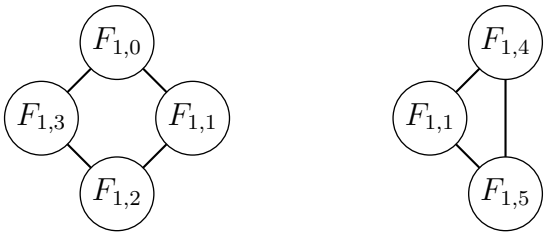
Voorbeeld 8. Neem uit verzameling F_2 twee elementen: $F_{2,0}, F_{2,1}$. Elk element is gedefinieerd door de ruimte omsloten door de 1-dimensionale lijnsegmenten, welke in een graaf weer te geven zijn.



Figuur 8.3: Grafische weergave van de twee vlakken.

Bij elk element is, zoals in fig. 8.4 een graaf op te stellen. Elke knoop staat voor een van de lijnstukken.

Door deze twee grafen te vergelijken, is te zien of de originele elementen aan elkaar grenzen. Elk overeenkomend facet in de grafen, is een facet waar de elementen



Figuur 8.4: Grafen van de vlakken uit fig. 8.3.

elkaar raken. Aan de bovenstaande grafen is te zien dat $F_{2,0}$ en $F_{2,1}$ samen $F_{1,1}$ delen. Dit komt overeen met de tekening in fig. 8.3, de vlakken delen één lijnstuk.

8.4 Efficiëntere representatie van een polytoop

Hoewel werken met de bovenstaande representatie door middel van een graaf snel kan zijn, omdat alle data expliciet aanwezig is, kost opslaan ervan veel ruimte. Dit vanwege de hoge hoeveelheid dubbele informatie die de grafen bevatten.

Tijdens het definiëren van een polytoop zijn grafen niet nodig, ze vormen slechts een tussenstap. We hebben net gezien dat twee knooppunten in een graaf door een rand verbonden worden, als de elementen die de knooppunten weergeven een facet delen. Om te weten uit welke facetten een element bestaat, is weer geen graaf nodig; een lijst met genummerde facetten is genoeg.

Aangezien ook dit recursief werkt, is een polytoop dus met de volgende onderdelen te definiëren: Een verzameling van coördinaten, en voor elke dimensie een verzameling, waarin elk element op zichzelf weer een verzameling is met de indexen van de facetten waaruit deze is opgebouwd.

Belangrijk om op te merken dat in de verzameling voor elk element er wordt verwezen naar de index van een facet van dat element in een lagere dimensie.

Voorbeeld 9. Dit is hoe de efficiëntere methode voor het opslaan gebruikt kan worden om een 3-kubus te definiëren in Python:

```
polytoop = {
    0: [[0,0,0], [0,0,1], [0,1,0], [0,1,1], [1,0,0],
       ↪ [1,0,1], [1,1,0], [1,1,1]], # vertexen
    1: [[0,1], [1,2], [2,3], [3,0], [4,5], [5,6], [6,7],
       ↪ [7,4], [0,4], [1,5], [2,6], [3,7]], # ribben
    2: [[0,1,2,3], [4,5,6,7], [0,9,4,8], [1,10,5,9],
       ↪ [2,11,6,10], [3,8,7,11]], # vlakken
    3: [[0,1,2,3,4,5]] # cellen
}
```

Hoofdstuk 9

Vinden van de optimale spanning tree

De volgende stap in het ontwikkelen van een polytoop is het vinden van de spanning tree van de facet-adjacentiegraaf van de polytoop. Dit proces is cruciaal om een gestructureerde en verbonden representatie van het polytoop te verkrijgen die kan worden uitgeklapt tot een netstructuur.

9.1 Waarom de methode resulteert in een netstructuur

Een netstructuur van een polytoop wordt gedefinieerd als een verzameling van de $(n-1)$-dimensionale facetten die volledig ontwikkeld zijn in een $(n-1)$-dimensionale ruimte, waarbij de relatieve verbindingen tussen aangrenzende facetten behouden blijven. De methode van het losmaken en positioneren van facetten voldoet aan deze definitie, omdat:

- Elk facet wordt losgesneden en afzonderlijk gepositioneerd in de lagere dimensie, wat garandeert dat alle facetten aanwezig zijn in de representatie.

- De verbindingen tussen aangrenzende facetten worden bepaald door het facet-adjacentiegraaf van de polytoop. Door een spanning tree van deze graaf te vinden, kunnen we zorgen dat elk facet direct of indirect verbonden blijft met elk ander facet.

- Een spanning tree zorgt ervoor dat er precies genoeg verbindingen tussen facetten zijn om de structuur verbonden te houden zonder extra cycli, wat consistent is met de definitie van een netstructuur.

9.2 Van graaf naar spanning tree

Het is relatief makkelijk om van een graaf, de bijbehorende spanning tree te creëren. Door simpelweg randen weg te halen en telkens te controleren of de resulterende graaf voldoet aan de eerder gestelde eisen, is een spanning tree te vinden. Een voorbeeld hiervan is te vinden in paragraaf 11.1.1.

9.3 Tree met minimale overlap

Het vinden van een spanning tree die overeenkomt met een netstructuur waarin de hoeveelheid overlap minimaal is, vormt een aanzienlijke uitdaging. Dit komt door de grote diversiteit aan polytopen en het feit dat het onmogelijk is om zonder expliciete constructie te bepalen of een spanning tree leidt tot een net zonder overlap. Het testen of een net daadwerkelijk vrij van overlap is, vereist altijd een concrete constructie en evaluatie.

Hoewel exacte oplossingen complex zijn, kunnen benaderingen voor overlapreductie worden toegepast. Een voorbeeld hiervan is beschreven in [39], waarin methoden worden besproken voor transformaties van $n = 3$ naar $n = 2$. Deze aanpak start met het toekennen van een gewicht aan elke rand in de graaf, gebaseerd op hun waarschijnlijkheid om overlap te veroorzaken. Vervolgens wordt de MST (minimal spanning tree) berekend, zodat de spanning tree de randen met de laagste gewichten bevat. Door problematische randen, die vanuit een overlappingsperspectief meer risico vormen, een hoger gewicht te geven, wordt de kans verkleind dat ze in de spanning tree worden opgenomen. Dit leidt tot een vermindering van overlap in de resulterende netstructuur.

9.3.1 Gewichten toekennen

Het ontwikkelen van effectieve heuristieken om gewichten toe te kennen aan randen op basis van de geometrische eigenschappen van de polytoop, is een uitdagend vraagstuk. Vanwege de complexiteit en omvang van deze analyses vallen de details hiervan buiten de reikwijdte van dit verslag. Een aantal goede heuristieken voor driedimensionale polytopen, wordt omschreven in *Creating optimized cut-out sheets for paper models from meshes*. Het ontwerpen van heuristieken voor polytopen met een hogere dimensionaliteit is echter een volledig nieuw onderzoeksgebied dat is voortgekomen uit ons onderzoek; er is hierover nog geen ander onderzoek gepubliceerd.

Hoofdstuk 10

Van spanning tree naar netstructuur

Het ogenschijnlijk eenvoudigste onderdeel van het algoritme is tegelijkertijd misschien wel het meest complex: het plaatsen van de facetten op basis van de spanning-tree. Alle facetten moeten zo aan elkaar geplaatst worden dat het facet-adjacentiegraaf van de resulterende netstructuur overeenkomt met de gecreëerde spanning tree.

Simpel gezegd moet elke facet verplaatst worden naar de lagere dimensie, en daarna naast een ander facet gezet worden volgens de spanning tree. Dit algoritmisch uitwerken, is echter heel ingewikkeld. We hebben het in meerdere stappen verdeeld, die voor elke facet worden gevolgd:

10.1 Facet verplaatsen naar lagere dimensie

Hoewel elke facet (vanuit de definitie) zich in een $(n-1)$-dimensionale affiene deelrumite bevind, wordt elke vertex alsnog door een n-dimensionale vector omschreven. Vanwege de definitie van een affiene deelrumite, weten we echter dat de gehele facet te verplaatsen is, zodat elke vertex met een $(n-1)$-dimensionale vector te omschrijven is. Dit is helaas niet triviaal.

10.1.1 Controleren van de dimensie van het facet

Hoewel het controleren van de dimensie in principe niet nodig is, kan het handig zijn om latere fouten te voorkomen, vooral bij een ongeldige invoer. Een facet bevindt zich in principe altijd in de affiene deelruimte met een dimensie één lager dan de polytoop. Controleren kan echter ook nooit kwaad.

In dit hoofdstuk zullen we niet verder in gaan op het algoritme om de dimensie van een affiene deelruimte te bepalen, omdat deze uitgebreid toegelicht staat in paragraaf 3.5 en appendix B.2.

10.1.2 Zwaartepunt van facet op oorsprong

Om de latere berekeningen makkelijker te maken, moet de polytoop P_n eerst verplaatst worden, zodat het zwaartepunt van het facet F_{n-1}, op de oorsprong $O = (0, 0)$ ligt. Dit kan gedaan worden door de gehele polytoop te transleren met een vector die de positie van het zwaartepunt naar de oorsprong verschuift. De translatieschaal is gebaseerd op het zwaartepunt van het facet, dat is het gemiddelde van de coördinaten van de betrokken hoekpunten $V(F_{n-1})$.

Laat $V_1, V_2, \ldots, V_k$ de hoekpunten van het facet F_{n-1} zijn. Het zwaartepunt G van deze facet wordt gegeven door:

$$G = \frac{1}{k} \sum_{i=1}^{k} V_i \tag{10.1}$$

waarbij k het aantal hoekpunten is van het facet. De verschuiving die nodig is om het zwaartepunt naar de oorsprong te verplaatsen, is de vector:

$$T = -G = -\frac{1}{k} \sum_{i=1}^{k} V_i \tag{10.2}$$

Deze translatieschaal T wordt vervolgens op elke hoekpunt V_i toegepast om de nieuwe positie van het facet te verkrijgen. Dit betekent dat de nieuwe positie van elk hoekpunt V_i' wordt gegeven door:

$$V_i' = V_i + T = V_i - \frac{1}{k} \sum_{i=1}^{k} V_i \tag{10.3}$$

Op deze manier wordt de gehele polytoop P_n zo verschoven dat het zwaartepunt van het facet F_{n-1} precies op de oorsprong ligt. Deze transformatie maakt de volgende berekeningen eenvoudiger, omdat we nu kunnen werken met een polytoop waarvan het zwaartepunt van het facet op de oorsprong is geplaatst.

10.1.3 Roteren naar een standaardvorm

Na het centreren van het facet op de oorsprong, is de volgende stap om de oriëntatie van de affiene deelruimte waarin het facet ligt, aan te passen. Dit wordt gedaan door de *Gram-Schmidt-orthonormalisatiemethode* toe te passen op de basisvectoren van de deelruimte. Hiermee roteren we de ruimte zodanig dat deze samenvalt met de standaardvorm $x_n = 0$.

Basisvectoren bepalen

Om de Gram-Schmidt-methode toe te passen, bepalen we eerst een basis voor de affiene deelruimte waarin het facet F_{n-1} zich bevindt. Laten we aannemen dat het

facet wordt beschreven door k hoekpunten, en dat deze hoekpunten zich na translatie op een as op de oorsprong bevinden. We kunnen dan de vectoren tussen een van de hoekpunten (bijvoorbeeld V_1) en de andere hoekpunten gebruiken om een set richtingsvectoren te bepalen:

$$\mathbf{b}_i = V_{i+1} - V_1, \quad i = 1, \ldots, k-1$$

waarbij $k-1$ het aantal onafhankelijke richtingsvectoren is in de affiene deelruimte. Deze vectoren bepalen een basis voor de deelruimte.

Gram-Schmidt-orthonormalisatie

Met de richtingsvectoren $\mathbf{b}_1, \mathbf{b}_2, \ldots, \mathbf{b}_{k-1}$ voeren we de Gram-Schmidt-methode uit om een orthonormale basis te verkrijgen. Het proces is als volgt:

1. Stel de eerste orthonormale vector gelijk aan de genormaliseerde $\mathbf{b}_1$:

$$\mathbf{u}_1 = \frac{\mathbf{b}_1}{|\mathbf{b}_1|}$$

2. Voor elke volgende vector $\mathbf{b}_i$, verwijder de projectie op de reeds georiënteerde vectoren $\mathbf{u}_1, \ldots, \mathbf{u}_{i-1}$, en normaliseer:

$$\mathbf{u}_i = \frac{\mathbf{b}_i - \sum_{j=1}^{i-1} \text{proj}_{\mathbf{u}_j}(\mathbf{b}_i)}{\left|\mathbf{b}_i - \sum_{j=1}^{i-1} \text{proj}_{\mathbf{u}_j}(\mathbf{b}_i)\right|}$$

waarbij de projectie $\text{proj}_{\mathbf{u}_j}(\mathbf{b}_i)$ wordt gegeven door:

$$\text{proj}_{\mathbf{u}_j}(\mathbf{b}_i) = \frac{\mathbf{b}_i \cdot \mathbf{u}_j}{\mathbf{u}_j \cdot \mathbf{u}_j}\mathbf{u}_j$$

Na dit proces verkrijgen we een orthonormale basis $\{\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_{k-1}\}$ voor de deelruimte.

Rotatiematrix construeren

Om de deelruimte waarin het facet ligt te roteren naar de standaardvorm $x_n = 0$, breiden we de orthonormale basis $\{\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_{k-1}\}$ uit tot een volledige basis voor $\mathbb{R}^n$. Dit doen we door een vector $\mathbf{u}_n$ te vinden die orthogonaal is aan de deelruimte en deze toe te voegen aan de basis.

De rotatiematrix $\mathbf{R}$ wordt vervolgens geconstrueerd door de basisvectoren als

kolommen te nemen:

$$\mathbf{R} = \begin{bmatrix} | & | & & | \\ \mathbf{u}_1 & \mathbf{u}_2 & \cdots & \mathbf{u}_n \\ | & | & & | \end{bmatrix}$$

Rotatie toepassen op het facet

Om de rotatie toe te passen, vermenigvuldigen we de rotatiematrix $\mathbf{R}$ met elke hoekpunt van het facet:

$$V_i' = \mathbf{R}V_i, \quad i = 1, \ldots, k$$

Na de rotatie liggen alle hoekpunten van het facet in de standaardvorm, waarbij de laatste coördinaat $x_n = 0$ is.

10.1.4 Projectie naar lagere dimensie

Na de rotatie, waarbij het facet zich nu volledig bevindt in de hypervlak $x_n = 0$, kunnen we de laatste coördinaat verwijderen. Dit reduceert de beschrijving van elk hoekpunt tot $(n-1)$-dimensionale coördinaten:

$$V_i'' = (x_1, x_2, \ldots, x_{n-1}), \quad i = 1, \ldots, k$$

Deze coördinaten beschrijven het facet in de lagere dimensie, zonder informatieverlies.

10.2 Facet verplaatsen naar andere facet

Nu moeten de facetten slechts nog verplaatst worden, zodat ze aan minstens één andere facet grenzen, volgens de gegenereerde spanning tree. Omdat we werken in $(n-1)$-dimensies, zijn we verzekerd dat deze structuur een valide netstructuur vormt.

We mogen op het facet basistransformaties toepassen: verplaatsen, roteren en spiegelen. Door deze transformaties blijft het facet zijn eigenschappen behouden. Spiegelen lijkt hier niet aan te voldoen, maar spiegelen komt overeen met rotatie in een hogere dimensie en dit is toegestaan.

Het is nu slechts nog een kwestie van het aflezen uit de grafen welke *sub-facetten* van de facetten aan elkaar moeten grenzen. Door de coördinaten van de hoekpunten identiek te maken volgens de bovenstaande basistransformaties, wordt het facet verplaatst naar de andere facet.

Door deze stappen voor elke facet te herhalen, ontstaat de $(n-1)$-dimensionale netstructuur.

Hoofdstuk 11

Casestudy deel 2: Het ontwikkelen van de n-kubus

In dit deel van de casestudy passen we de eerder ontworpen algoritmen toe op verschillende n-kubussen. Hiermee maken we de omschrijving van het algoritme minder abstract en hopen we dat het voor iedereen beter toepasbaar is.

11.1 De 3-kubus

We beginnen niet met de simpelste variant, maar wel met de variant waar de meeste van ons het meest bekend mee zijn: een 'normale' kubus. We hebben op de basisschool allemaal minstens één kubus gemaakt vanuit een netstructuur, dus weten al hoe deze uitslag er uit komt te zien.

Toch volgen we in deze casestudy het algoritme van begin tot eind, dan kunnen we daarna bekijken of we een kloppende uitslag hebben gekregen.

Representatie polytoop als een graaf

De eerste stap is het weergeven van de kubus als een graaf. We volgen in dit voorbeeld onze stappen achteruit, omdat dit beter te volgen is.

Hoekpunten

In paragraaf 6.4 hebben we gezien dat elk coördinaat van een hoekpunt van de n-kubus te vinden is door de index om te zetten naar een binair getal met n bits. Het hoekpunt met index 0, heeft dus een coördinaat van $(0, 0, 0)$. Hoekpunt 1 heeft coördinaat $(0, 0, 1)$, en bij hoekpunt 7 hoort bijvoorbeeld de coördinaat $(1, 1, 0)$.

Bovendien weten we (vanwege verg. (6.4)) dat een 3-kubus $2^3 = 8$ hoekpunten

heeft. Hiermee kunnen we de verzameling van hoekpunten betekenis geven:

$$F_0 = \{F_{0,0}, F_{0,1}, F_{0,2}, \ldots, F_{0,7}\} \tag{11.1}$$

Opmerking 5. De verzameling hoekpunten loopt tot $F_{0,7}$, dit omdat we zijn begonnen met tellen bij index 0.

Deze verzameling is ook te schrijven als een verzameling coördinaten of vectoren:

$$F_0 = \left\{ \begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\0\\1\end{pmatrix}, \begin{pmatrix}0\\1\\0\end{pmatrix}, \begin{pmatrix}0\\1\\1\end{pmatrix}, \begin{pmatrix}1\\0\\0\end{pmatrix}, \begin{pmatrix}1\\0\\1\end{pmatrix}, \begin{pmatrix}1\\1\\0\end{pmatrix}, \begin{pmatrix}1\\1\\1\end{pmatrix}, \right\} \tag{11.2}$$

Lijnstukken

Hierna moet de verzameling lijnstukken opgesteld worden, dit zijn er 12 (uit verg. (6.6)):

$$F_1 = \{F_{1,0}, F_{1,1}, \ldots, F_{1,11}\} \tag{11.3}$$

Elk lijnstuk wordt gegeven door een paar hoekpunten. De nummering maakt hierbij weinig uit, dus ze zijn in deze casestudy telkens genummerd zodat de som van de coördinaten van de hoekpunten oploopt. Dit komt ongeveer neer op: we beginnen bij hoekpunt 0, dan hoekpunt 1, enzovoort.

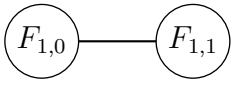
Een voorbeeld hiervan is $F_{1,0}$, het 1-dimensionale lijnstuk met index 0:

$$F_{1,0} = \{F_{0,0}, F_{0,1}\} = \left\{ \begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\0\\1\end{pmatrix} \right\}$$

$$F_{1,1} = \{F_{0,0}, F_{0,2}\} = \left\{ \begin{pmatrix}0\\0\\0\end{pmatrix}, \begin{pmatrix}0\\1\\0\end{pmatrix} \right\}$$

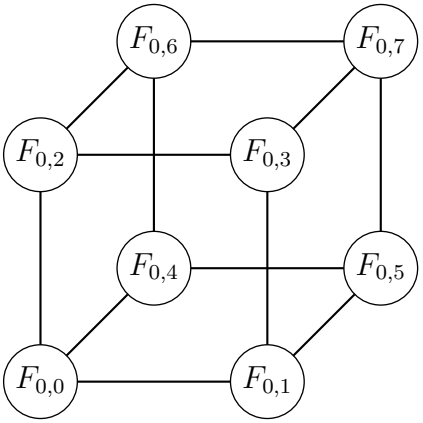
Opmerking 6. We hebben hier voor het gemak niet alle lijnstukken weergegeven. Maar als je meedoet, voel je vooral vrij om deze alsnog allemaal even op te schrijven.

Élk lijnstuk is weer te geven met een graaf, ook al zijn deze nu nog niet zo interessant. De graaf die correspondeert met $F_{1,0}$ ziet er bijvoorbeeld als volgt uit:



Figuur 11.1: $F_{1,0}$

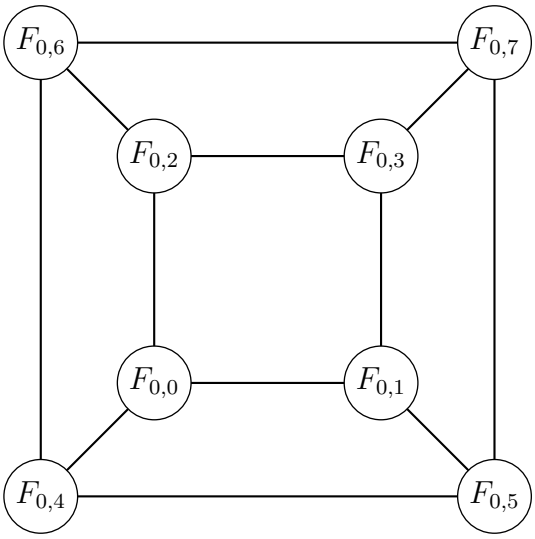
Interessanter wordt het als je al deze grafen zou combineren. Verbind elke knoop met alle andere knopen uit andere grafen waar deze mee verbonden is, er ontstaat een *samengestelde graaf*, zie fig. 11.2.



Figuur 11.2: De samengestelde graaf van alle lijnstukken

Deze graaf lijkt op een kubus, dat kan geen toeval zijn! Dat is het ook zeker niet, denk maar eens aan de voorwaarden waarop de knopen met elkaar zijn verbonden. Elke knoop staat voor een hoekpunt. En als twee hoekpunten door een lijnstuk worden verbonden, ontstaat er een rand tussen de knopen.

Daarnaast hebben we een klein beetje geholpen door de knopen op een specifieke manier neer te zetten. We hadden er ook voor kunnen kiezen om ze als volgt neer te zetten, dan was het een stuk minder duidelijk geweest:



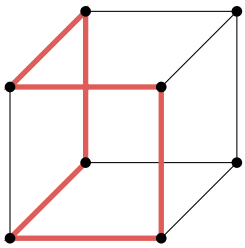
Figuur 11.3: De samengestelde graaf anders weergegeven

Vlakken

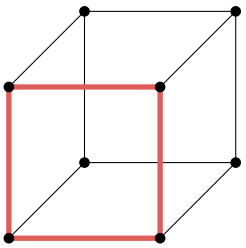
Op dezelfde manier is ook voor elk vlak een graaf op te stellen. De facet-adjacentiegraaf van een vlak bevat, zoals eerder aangetoond, alle zijdes die dat vlak omsluiten. Deze

zijdes hebben we eerder al omschreven met hun eigen grafen.

Ook hebben we kunnen zien op basis van welke voorwaarde twee lijnstukken aan elkaar grenzen. Dit is al ze een vertex delen. Om de vlakken te definiëren, moeten we dus kijken naar de lijnstukken, en met welke andere lijnstukken deze zijn verbonden. Hierdoor ontstaan er cykels, een cykel is een lus waarin lijnstukken onderling met elkaar verbonden zijn. Elk vlak hoort bij een cykel, dit omdat een vlak altijd gesloten is. Niet bij elke cykel hoort echter een vlak, zie de afbeeldingen hieronder.

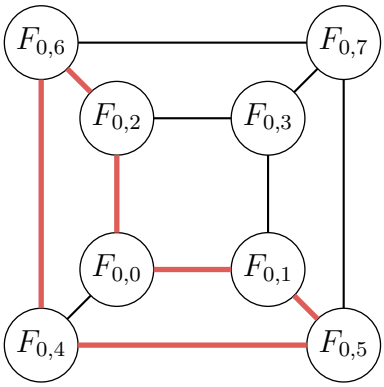


(a) Een cykel zonder enkel vlak

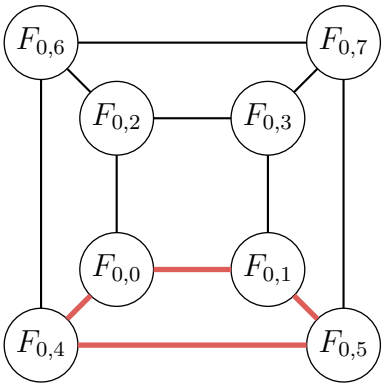


(b) Een cykel met vlak

Figuur 11.4: Cykels van ribben kunnen corresponderen aan vlakken



(a) Een cykel zonder enkel vlak



(b) Een cykel met vlak

Figuur 11.5: Cykels van ribben als grafen

Een cykel in de graaf correspondeert met een vlak als deze niet op te delen is in een of meerdere kleinere cykels, waarbij alle knopen alsnog verbonden zijn. Neem als voorbeeld fig. 11.5a. Deze cykel $\{F_{0,0}, F_{0,1}, F_{0,5}, F_{0,4}, F_{0,6}, F_{0,2}, F_{0,0}\}$ is op te delen in twee kleinere cykels waarbij elke knoop behouden blijft:

$$\{F_{0,0}, F_{0,1}, F_{0,5}, F_{0,4}, F_{0,0}\} \quad \text{en} \quad \{F_{0,0}, F_{0,2}, F_{0,6}, F_{0,4}, F_{0,0}\},$$

deze corresponderen beide wel met een vlak.

Nu elk vlak omschreven kan worden door een cykel, is voor elk vlak een facet-adjacentiegraaf op te stellen, op basis van de lijnstukken (de facetten van het vlak)

in de cykel. In deze graaf is elk lijnstuk een knoop, waarbij twee knopen verbonden zijn als de lijnstukken elkaar in een vertex raken.

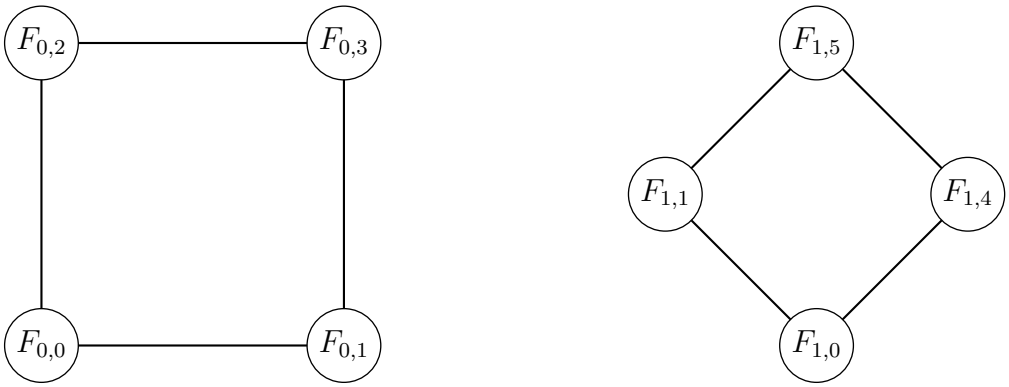
We laten hier als voorbeeld zien hoe we de graaf bij het vlak dat we $F_{2,0}$ hebben genoemd kunnen maken. Dit vlak bestaat uit de lijnstukken $F_{1,0}, F_{1,1}, F_{1,5}, F_{1,4}$, wat we ook kunnen noteren als

$$F_{1,0}, F_{1,1}, F_{1,5}, F_{1,4} \in F_{2,0} \tag{11.4}$$

De hoekpunten die bij vlak $F_{2,0}$ horen zijn $F_{0,0}, F_{0,1}, F_{0,3}, F_{0,2}$, wat te noteren is als

$$F_{0,0}, F_{0,1}, F_{0,3}, F_{0,2} \in F_{2,0} \tag{11.5}$$

In figuur 11.6 is dit vlak op twee manieren als graaf weergegeven. In afbeelding 11.6a gebeurt dit door alle knopen de hoekpunten van het vlak te maken, dit is een onderdeel van de samengestelde graaf in figuur 11.3. Hierin stelt elke verbinding een lijnstuk voor, figuur 11.6b geeft deze lijnstukken telkens als knoop weer. De rare nummering van deze lijnstukken komt door de eerder genoemde manier van lijnstukken nummeren.



(a) Knopen zijn hoekpunten (b) Knopen zijn lijnstukken

Figuur 11.6: Een vlak weergegeven als graaf

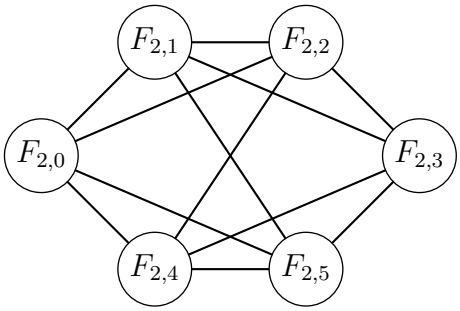
Te zien is dat elke knoop in de tweede graaf, in de eerste graaf een verbinding is. Dit komt doordat de knopen in fig. 11.6b verbonden zijn als de lijnstukken elkaar raken, en ze dus een vertex delen. Deze gedeelde vertexen zijn dan weer weergegeven in fig. 11.6a.

Ook nu is het mogelijk een samengestelde graaf op te stellen, maar dit keer voor de vlakken, dit laten we echter hier buiten beschouwing.

Gehele polytoop (cellen)

Het volgende element zijn de 3-dimensionale cellen. Omdat onze originele polytoop echter ook 3-dimensionaal was, spreken we hier eerder van de gehele polytoop. Voor

de polytoop is ook een facet-adjacentiegraaf op te stellen, waarbij de facetten de 2-dimensionale vlakken zijn. Omdat een n-dimensionale polytoop per definitie slechts uit één n-dimensionaal element bestaat (zie verg. (6.7)). Er is dus nu ook maar één graaf op te stellen (fig. 11.7).

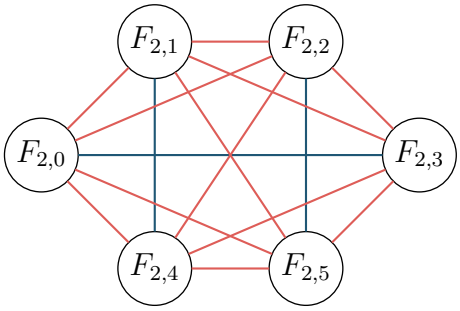


Figuur 11.7: De graaf van de 3-kubus

Deze graaf bevat een aantal interessante eigenschappen. De belangrijkste is dat de knopen in de graaf paren vormen. Deze paren zijn onderling niet verbonden, maar de knopen zijn verder aan alle andere knopen verbonden. Dit volgt uit dat de graaf laat zien welke vlakken aan elkaar grenzen. Elk vlak op een kubus staat tegenover één ander vlak en deze raken elkaar niet. Zo is $F_{2,0}$ niet verbonden met $F_{2,3}$.

Deze eigenschap blijft behouden bij de n-kubus in hogere dimensies en is een direct gevolg van de definitie van de n-kubus. De facet-adjacentiegraaf voor elke n-kubus wordt een *complementaire graaf van een perfecte matchingsgraaf* genoemd. Een perfecte matchingsgraaf is een graaf waarbij elke knoop verbonden is met exact één andere knoop. De complementaire graaf geeft aan dat het tegenovergestelde hiervan is genomen, waarbij alle bestaande verbindingen verdwijnen en op alle plekken waar geen verbinding was, een verbinding verschijnt.

In fig. 11.8 is in blauw de perfecte matchingsgraaf en in rood de complementaire graaf weergegeven.



Figuur 11.8: Een complementaire graaf van een perfecte matchingsgraaf

Met deze eigenschap zijn veel interessante dingen te doen, maar deze vallen helaas buiten de focus van dit profielwerkstuk. Wij denken echter dat deze eigenschap het

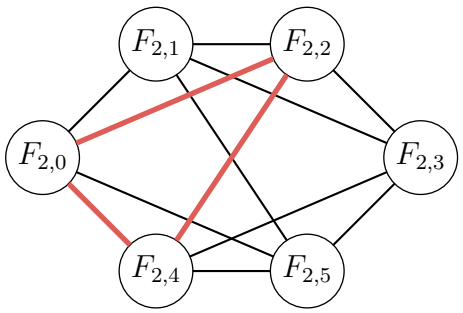
beginpunt kan zijn voor het oplossen van een van de problemen die we benoemen in de discussie.

Ook kan dit een makkelijke manier zijn om te bepalen welke structuur de spanning tree van een n-kubus heeft, omdat het bepalen van de facet-adjacentiegraaf een stuk simpeler is.

11.1.1 Vinden van de spanning tree

De volgende stap in het ontwikkelen van de 3-kubus is het vinden van de spanning tree bij de facet-adjacentiegraaf.

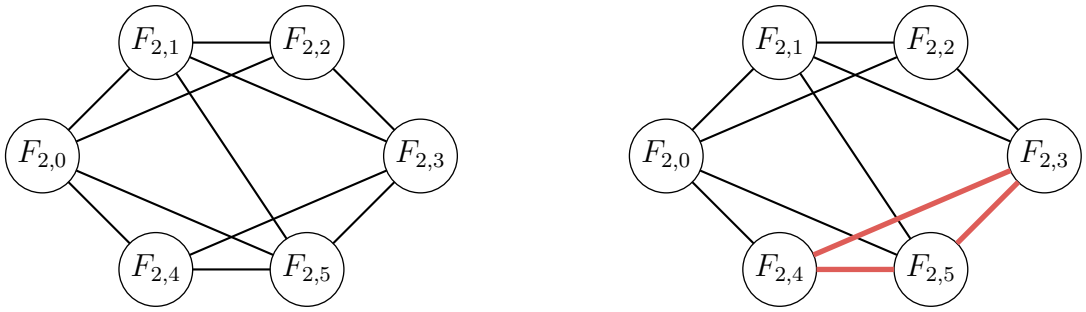
Er zijn veel manieren om dit te doen, maar we leggen hier de focus op de eenvoudigste: het stap voor stap verwijderen van lijnstukken. Als na het verwijderen van het lijnstuk wordt gecontroleerd of de graaf aan de voorwaarden van een spanning tree, ontstaat deze tree uiteindelijk.



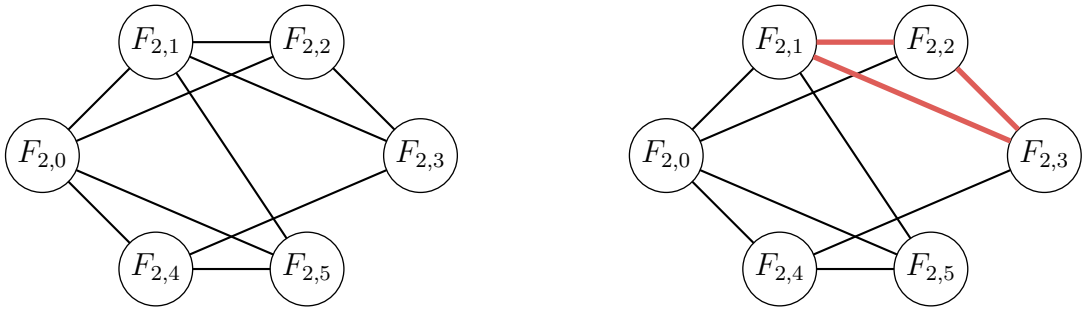
Figuur 11.9: Graaf met een cykel

We beginnen met de graaf zoals in het vorige onderdeel beschreven. Deze is nogmaals afgebeeld in fig. 11.9. Het is nu zaak stap-voor-stap lijnstukken te verwijderen, zodat er geen cykels meer zijn. Eerst is het belangrijk te controleren of de graaf niet al een spanning tree is, maar vanwege bijvoorbeeld de cykel die rood gemarkeerd is, is dit niet het geval. De gemarkeerde cykel is niet de enige cykel die zich in de graaf bevindt, maar dit is niet relevant. Het is de eerste cykel die we hebben gevonden, dus we kunnen meteen door naar de volgende stap.

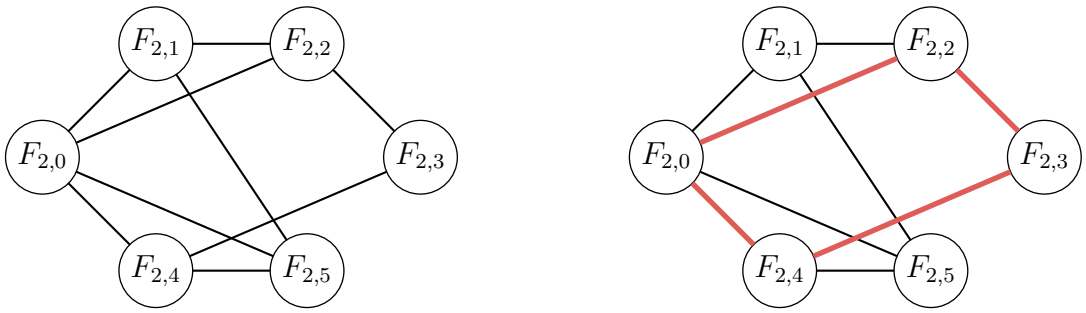
Als er in een graaf een cykel gevonden is, is de efficiëntste manier om er een spanning tree van te maken, een van de randen van de cykel weg te halen. Wat er nu ontstaat, is in fig. 11.10 weergeven. Helaas bevat de graaf nog steeds een of meerdere cykels, een hiervan is ook dit figuur weergegeven.



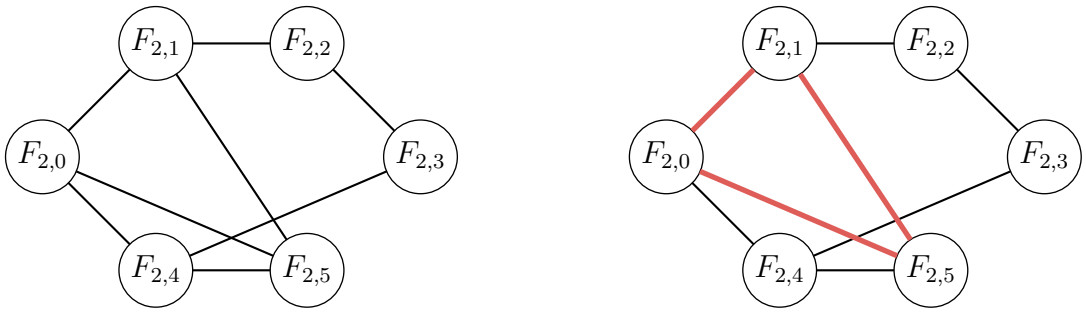
Figuur 11.10: Rand $F_{2,2}F_{2,4}$ verwijderd



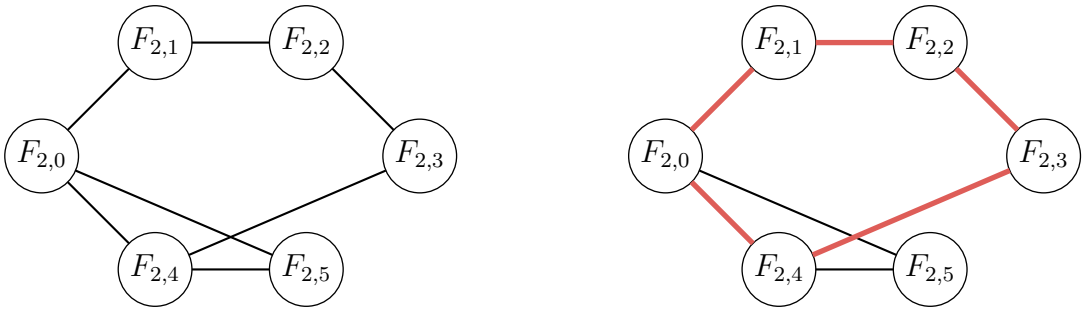
Figuur 11.11: Rand $F_{2,3}F_{2,5}$ verwijderd



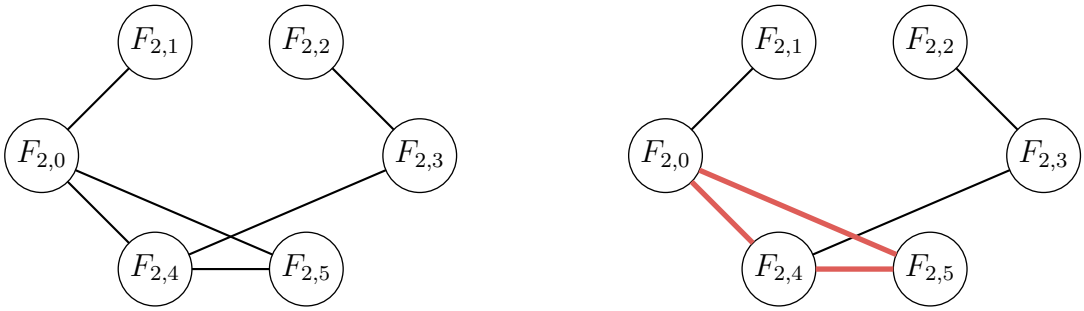
Figuur 11.12: Rand $F_{2,1}F_{2,3}$ verwijderd



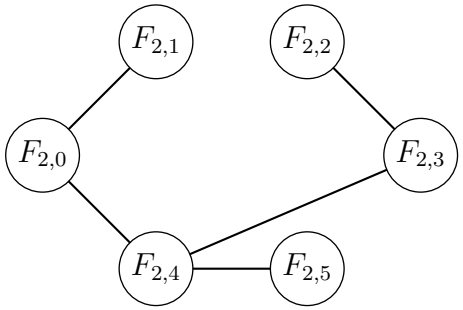
Figuur 11.13: Rand $F_{2,0}F_{2,2}$ verwijderd



Figuur 11.14: Rand $F_{2,1}F_{2,5}$ verwijderd

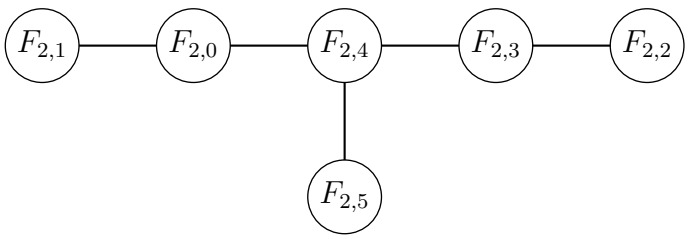


Figuur 11.15: Rand $F_{2,1}F_{2,2}$ verwijderd



Figuur 11.16: Rand $F_{2,0}F_{2,5}$ verwijderd

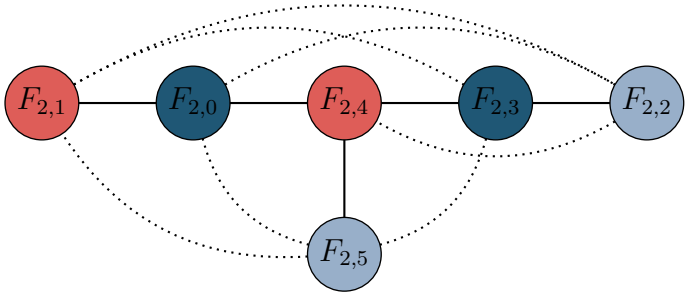
De ontstaande graaf (fig. 11.16) is de spanning tree van de originele graaf. Deze is wel nog duidelijker weer te geven, zoals in figuur fig. 11.17.



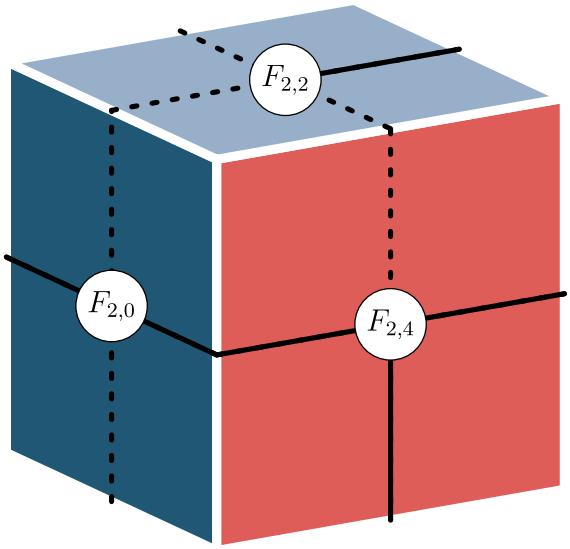
Figuur 11.17: Spanning tree duidelijker weergegeven

Door deze tree over de originele 3-kubus neer te leggen, is te zien over welke lijnstukken de kubus *opengesneden* moet worden. In figuren 11.18 en 11.19 is de

spanning tree met een normale lijn weergegeven. Alle stippellijnen geven een verbinding aan die in de netstructuur wordt weggehaald.



Figuur 11.18: Spanning tree ingekleurd



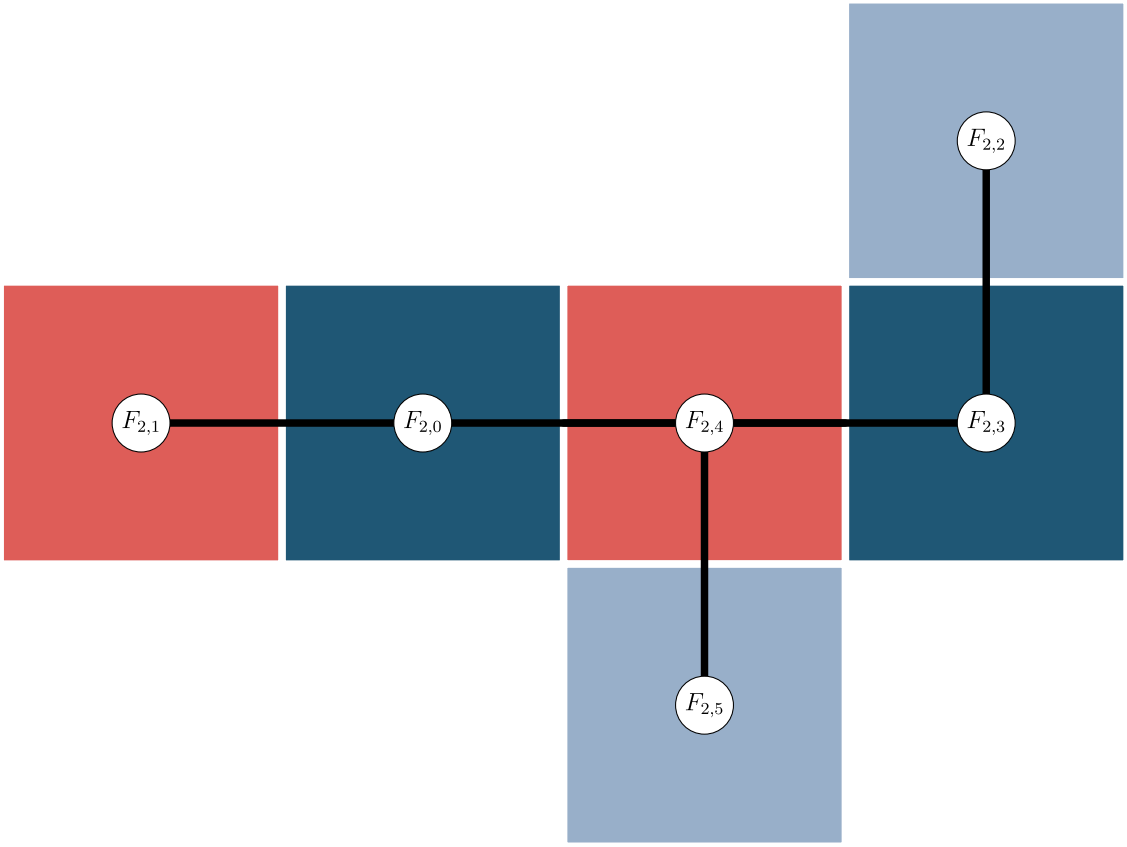
Figuur 11.19: De spanning tree weergegeven op de 3-kubus

Het is belangrijk te beseffen dat elke verbinding in de graaf, ook staat voor een lijnstuk in de kubus. Dit komt doordat er verbindingen ontstaan omdat twee vlakken een gemeenschappelijk lijnstuk hebben. Hierdoor is elke verbinding inherent verbonden met een lijnstuk.

De facetten van de kubus zijn nu te verplaatsen, zodat deze aan elkaar grenzen volgens de spanning tree. We laten de details hiervan hier buiten wegen, omdat deze zonder verdere uitleg goed te begrijpen zijn.

Figuur 11.20 geeft de netstructuur die bij de spanning tree hoort weer. De knik in de netstructuur wordt veroorzaakt door welke lijnstukken er tussen vlakken worden gedeeld.

We hebben dus uitgewerkt hoe een 3-dimensionale kubus te ontwikkelen is naar een 2-dimensionale netstructuur.



Figuur 11.20: Het net en de spanning tree van de 3-kubus

11.2 De 7-kubus

In dit onderdeel onderzoeken we kort de stappen voor het ontwikkelen van een 7-dimensionale kubus, ofwel de 7-kubus. Deze objecten bestaan uit complexe geometrieën, die moeilijk te visualiseren zijn vanwege het hoge aantal dimensies. Omdat een gedetailleerde uitwerking van dit proces veel ruimte in beslag zou nemen, hebben we besloten de uitleg te vereenvoudigen, waarbij we ons focussen op de belangrijkste stappen en eigenschappen van de kubus.

De basisstructuur van een n-dimensies kubus wordt bepaald door zijn facetten en de verbindingen tussen deze facetten. In het geval van de 7-kubus spreken we van een hyperkubus in zeven dimensies, wat betekent dat het object bestaat uit 14 facetten (elk facet is een 6-kubus) en 128 knopen (deze vertegenwoordigen de hoeken van de kubus). Deze facetten zijn onderling verbonden volgens specifieke regels die bepalen hoe de geometrie van de kubus zich uitstrekt in de zevende dimensie.

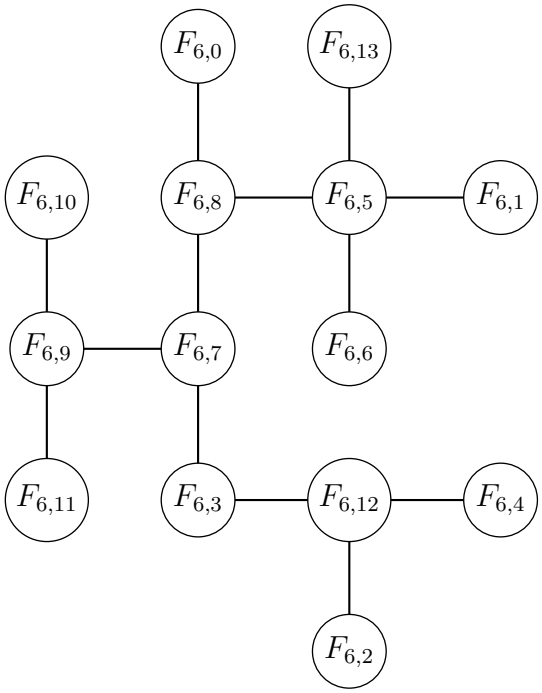
Zoals eerder besproken, is elke facet-adjacentiegraaf van een n-kubus een complementaire graaf van een perfecte matchingsgraaf, met $n \times 2$ knopen. Deze graaf is een belangrijk hulpmiddel bij het begrijpen van de onderlinge relaties tussen de facetten van de kubus. De grafen voor hogere-dimensionale kubussen (zoals de 7-kubus) worden vaak weergegeven in een 2D-projectie, wat het visueel complex maakt om

alle onderlinge verbindingen te volgen.

Een visuele representatie van deze graaf is te vinden op de achterkant van dit profielwerkstuk, waarin de knopen en hun onderlinge verbindingen duidelijk worden weergegeven. Bovendien is een mogelijke spanning tree van de graaf gemarkeerd.

Bij het bekijken van de graaf valt op dat bepaalde paren knopen, zoals $F_{6,0}$ en $V_{6,7}$, niet met elkaar zijn verbonden. Dit is belangrijk omdat ze ons laten zien welke facetten van de kubus afhankelijk zijn van andere facetten en welke mogelijk onafhankelijk zijn, wat belangrijk is voor het uitvouwen van de polytoop.

De spanning tree is op een duidelijkere manier weergegeven in figuur 11.21. De verbindingen in de spanning tree representeren de minimale set van verbindingen die nodig zijn om alle facetten van de kubus met elkaar te verbinden, zonder dat er onnodige verbindingen zijn.



Figuur 11.21: Spanning tree van de 7-kubus

Door de spanning tree over de 7-kubus te leggen, ontstaat de netstructuur. Maar omdat we de zevende dimensie niet kunnen visualiseren, biedt dit ons weinig houvast. Ook het weergeven van de 6-dimensionale netstructuur is geen eenvoudige opgave, waardoor we deze eveneens buiten beschouwing laten. Daarom hebben we ervoor gekozen om niet alle 128 vertexen te visualiseren, maar ons te beperken tot slechts de facet-adjacentiegraaf. Theoretisch gezien is de 7-kubus hiermee uitgevouwen tot een netstructuur—al zal een wezen in een 7-dimensionaal universum nog de nodige stappen moeten zetten om deze daadwerkelijk te bouwen.

Deel III

Afsluiting

Hoofdstuk 12

Discussie & Conclusie

In dit profielwerkstuk hebben we onderzocht hoe n-dimensionale polytopen kunnen worden ontwikkeld tot $(n-1)$-dimensionale netstructuren. We stelden ons het doel om een systematische methode te ontwikkelen die, ongeacht de vorm van een polytoop, deze naar een lagere dimensie kan ontwikkelen. Terugkijkend op het gehele proces kunnen we met tevredenheid concluderen dat we deze doelstelling hebben bereikt. We hebben niet alleen een theoretisch kader opgebouwd rond polytopen en hun ontwikkeling, maar ook een praktisch toepasbaar algoritme gecreëerd dat we succesvol hebben getest en dat, omdat het zich volledig baseert op de fundamentele definities en eigenschappen die elke polytoop bezit, in principe voor iedere polytoop toepasbaar is.

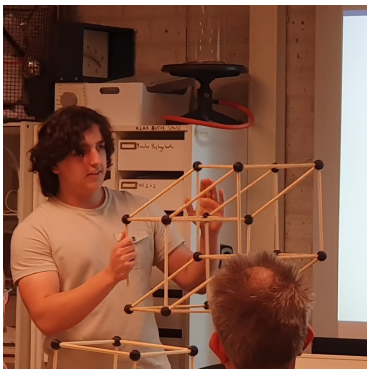
12.1 De educatieve waarde van ons onderzoek

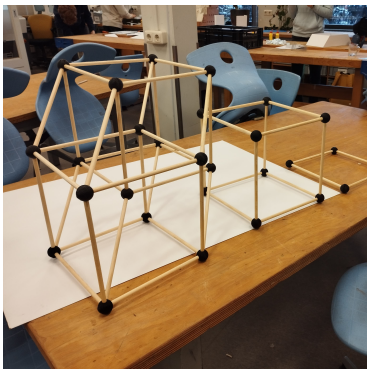
Een van de belangrijkste doelstellingen die we in de inleiding formuleerden, was om op een toegankelijke wijze inzicht te bieden in hogere dimensies en aan te tonen hoe de abstracte wiskunde van polytopen en netstructuren intuïtief kan worden begrepen. Het feit dat u, als lezer, dit gehele verslag heeft doorgelezen, is voor ons een belangrijke indicatie dat we in dit doel zijn geslaagd. We hebben de materie zodanig weten te structureren en te presenteren dat deze toegankelijk is geworden voor een breed publiek.

Deze educatieve waarde is verder bevestigd door onze ervaringen met het delen van onze bevindingen. Tijdens het onderzoek hebben we de kans gekregen om lezingen en workshops te geven aan diverse groepen, van basisschoolleerlingen tot professionals. De positieve reacties die we ontvingen, hebben ons ervan overtuigd dat het visualiseren van hogere dimensies via netstructuren een goed educatief hulpmiddel is. Het stelde ons in staat om abstracte wiskundige concepten op een intuïtieve en visueel aantrekkelijke manier over te brengen.

Voor degenen die geïnteresseerd zijn in het verder verkennen van dit onderwerp,

hebben we een website opgezet (`pws.bramleisink.nl`) waar aanvullende visualisaties, demonstraties en onderwijsmateriaal te vinden zijn. Deze website dient als een verlengstuk van dit profielwerkstuk en onderstreept ons streven om hogere-dimensionale wiskunde toegankelijker te maken voor een breed publiek. We zijn van plan de inhoud van de website in de toekomst verder aan te vullen met nieuwe materialen en updates.





12.2 Beperkingen van de theorie en methoden

Ondanks onze successen kent onze aanpak ook beperkingen. Het visualiseren van objecten uit hogere dimensies blijft een uitdaging, aangezien we afhankelijk zijn van projecties en wiskundige modellen. Hoewel we ons best doen om deze concepten toegankelijk te maken, blijft het voor sommige mensen moeilijk om een volledig intuïtief beeld te vormen. Het overbruggen van deze kloof is een uitdaging, maar we blijven werken aan nieuwe manieren om dit beter te realiseren.

Ook de toepassing van grafentheorie heeft zijn grenzen. Het vertalen van grafen naar netstructuren in hogere dimensies wordt gecompliceerd door het feit dat niet elke polytoop noodzakelijkerwijs een overlapvrij net kan hebben. Hoewel we het toestaan van overlap als oplossing hebben overwogen, blijft het voor veel polytopen onduidelijk of een niet-overlappend net altijd mogelijk is. Voor 3D-polytopen biedt de Stelling van Alexandrov enige houvast, maar in hogere dimensies is dit nog een open vraag.

12.3 De casestudy van de n-kubus

De n-kubus heeft als casestudy waardevolle inzichten opgeleverd. Het proces van het ontwikkelen naar een $(n-1)$-dimensionaal net demonstreert hoe hogere-dimensionale structuren visueel benaderd kunnen worden. Dit heeft niet alleen geleid tot interessante grafische representaties, maar ook tot een dieper begrip van de onderliggende structuur dankzij de toepassing van spanning trees en grafentheorie.

We hebben echter ook geconstateerd dat de complexiteit exponentieel toeneemt naarmate de dimensie groeit, waardoor de huidige methoden steeds rekenintensievere worden. Dit benadrukt het belang van het ontwikkelen van efficiëntere algoritmen en heuristieken voor het werken met hogere dimensies.

12.4 Toepassingen in de praktijk

Zoals we in de inleiding al beschreven, zijn de toepassingen van dit onderzoek breed en relevant. In de computergraphics kan onze aanpak helpen bij het visualiseren van hogere-dimensionale data, cruciaal voor gebieden als machine learning en kunstmatige intelligentie. Netstructuren kunnen complexe datastructuren vereenvoudigen en toegankelijker maken voor analyse.

In de natuurkunde hebben onze methoden toepassingen in onder andere kristallografie, moleculaire modellering en zelfs snaartheorie. Hier spelen hogere-dimensionale structuren een fundamentele rol, en onze benadering kan bijdragen aan het begrijpen van de symmetrieën van ruimte-tijd. Dit werk legt daarmee een solide basis voor verder onderzoek binnen zowel de wiskunde als de natuurkunde, precies zoals we in onze doelstelling beoogden.

Ook in meer praktische toepassingen, zoals we noemden in onze inleiding, blijkt de waarde van ons onderzoek. Van verpakkingsontwerp tot biomedische wetenschap, het concept van het ontwikkelen van objecten naar een lagere dimensie biedt verrassend veel praktische toepassingsmogelijkheden.

12.5 Toekomstig onderzoek

Er zijn veel interessante richtingen voor verder onderzoek. Het optimaliseren van algoritmen voor het uitvouwen van polytopen is een grote uitdaging, vooral in hogere dimensies waar de berekeningen complexer worden.

Een ander interessant onderwerp is het vinden van niet-overlappende netstructuren. Het is nog steeds onduidelijk of dit voor elke convexe n-dimensionale polytoop mogelijk is. Dit maakt het een fascinerend onderzoeksgebied. Ook het ontwikkelen van methoden om hogere-dimensionale polytopen te analyseren zonder dat belangrijke geometrische eigenschappen verloren gaan, zou een enorme stap voorwaarts zijn.

Verder zien we veel potentieel in toepassingen buiten de wiskunde, zoals kunstmatige intelligentie. Netstructuren kunnen een rol spelen in het visualiseren van multidimensionale data en het verbeteren van methoden voor clustering, patroonherkenning en dimensiereductie. Dit biedt volop mogelijkheden voor innovatie.

Tijdens dit profielwerkstuk hebben we onszelf veel vragen gesteld. Sommige vragen hebben we opgelost en in dit verslag opgenomen, maar er blijven ook een aantal interessante vraagstukken open. Hier zijn vijf vragen waar we graag een antwoord op zouden zien:

1. Is elke polytoop te definiëren door slechts hoekpunten en hun relaties?
2. Is er een functie $f(n)$ die het aantal unieke netstructuren van een n-kubus beschrijft?
3. Is elke convexe n-dimensionale polytoop uit te vouwen naar een netstructuur zonder overlap?
4. Zijn er, naast de tetraëder, andere polytopen die een netstructuur met een convexe buitenrand kunnen hebben?
5. Wat zijn heuristieken om overlap in hoger-dimensionale netstructuren te voorkomen?

Deze vragen vormen een mooie basis voor toekomstig onderzoek. Het beantwoorden ervan zou niet alleen onze kennis van polytopen verdiepen, maar zou ons ook persoonlijk veel plezier geven. Om deze reden is elk vraagstuk verder uitgewerkt in appendix C.

12.6 Conclusie

Dit profielwerkstuk heeft ons in staat gesteld om de doelen die we in de inleiding formuleerden te realiseren. We hebben een toegankelijke methodologie ontworpen voor het ontwikkelen van n-dimensionale polytopen naar $(n-1)$-dimensionale netstructuren, en we hebben aangetoond hoe abstracte wiskunde intuïtief kan worden begrepen door middel van visualisatie en praktische toepassingen.

Door grafentheorie en de wiskundige eigenschappen van polytopen te combineren, hebben we een methode gepresenteerd die niet alleen theoretisch onderbouwd is, maar ook praktisch toepasbaar blijkt te zijn. Het educatieve aspect van ons werk, versterkt door onze presentaties en online materialen, heeft bijgedragen aan het vergroten van het begrip en de waardering voor hogere-dimensionale wiskunde bij diverse doelgroepen.

Hoewel er nog uitdagingen blijven bestaan, zoals het verbeteren van algoritmen en het onderzoeken van niet-overlappende netstructuren, biedt dit werk een solide fundament voor toekomstig onderzoek. Op het kruispunt van multidimensionale meetkunde, grafentheorie en computerwetenschap hebben we niet alleen een theoretisch interessant, maar ook een praktisch relevant onderzoeksgebied verkend - precies zoals we ons hadden voorgenomen.

Woordenlijst

n-kubus De generalisatie van een kubus naar n dimensies, bijvoorbeeld een 3-kubus (3D) of een tesseract (4D). 14

n-tuple Een geordende verzameling van n waarden, bijvoorbeeld om de coördinaten van een punt in n dimensies te beschrijven. 21

4D-ruimtetijd Een concept dat ruimte (3 dimensies) en tijd (1 dimensie) combineert tot een vierdimensionale structuur, zoals gebruikt in de relativiteitstheorie. 19

affiene deelruimte Een lineaire deelruimte die niet noodzakelijk door de oorsprong gaat, maar wordt verschoven door een vector. 36

as Een rechte lijn in een coördinatenstelsel waarmee posities worden beschreven, zoals de x-as of y-as. 20

binair Een systeem of voorstelling gebaseerd op twee mogelijke waarden, meestal 0 en 1, gebruikt in bijvoorbeeld logica en computerwetenschappen. 61

bowlingbal Een driedimensionale bol met een glad oppervlak en vaak drie gaten, gebruikt in het bowlen. 37

cartesisch coördinatenstelsel Een assenstelsel waarin posities worden beschreven met coördinaten ten opzichte van loodrechte assen. 20, 25

cel Een 3D-component van een polytoop in hogere dimensies, bijvoorbeeld een kubus in een tesseract. 32

combinatoriek De tak van de wiskunde die zich richt op het tellen, ordenen en combineren van objecten volgens bepaalde regels. 59

complementaire graaf Een graaf die dezelfde knopen heeft als de oorspronkelijke graaf, maar waarbij twee knopen precies dan verbonden zijn als ze in de oorspronkelijke graaf niet verbonden zijn. 89

convexe vorm Een geometrische vorm waarbij een lijnsegment tussen twee willekeurige punten volledig binnen de vorm ligt. 14

coördinaat Een getal dat de positie van een punt langs een as in een ruimte aangeeft. 21

coördinatenvlak Een vlak in een cartesisch coördinatenstelsel dat wordt gevormd door twee assen, zoals de x-y-vlak. 20

deelruimte Een subruimte binnen een grotere vectorruimte, bijvoorbeeld een vlak in een 3D-ruimte. 37

dimensie Een onafhankelijke richting waarin een object of ruimte kan worden uitgestrekt, bijvoorbeeld lengte, breedte en hoogte. 14

echter Ons favoriete voegwoord. 13

euclidische afstand De kortste afstand tussen twee punten in een euclidische ruimte, gemeten langs een rechte lijn. 26

euclidische ruimte Een meetkundige ruimte waarin de regels van de euclidische meetkunde gelden, zoals rechte lijnen en vlakken. 19

facet De $n-1$-dimensionale bouwstenen van een polytoop bijvoorbeeld een vlak in een veelvlak. 32

facet-adjacentiegraaf Een graaf die de onderlinge relaties tussen de facetten van een polytoop toont, waarbij elke knoop een facet is en er een verbinding is tussen twee facetten als ze een gemeenschappelijke rand delen. 75

gerichte graaf Een graaf waarbij elke verbinding een richting heeft, weergegeven door pijlen tussen knopen. 46

getallenlijn Een visuele representatie van reële getallen op een rechte lijn. 20

gewogen graaf Een graaf waarbij elke verbinding een gewicht of waarde heeft, bijvoorbeeld een afstand of kosten. 47

graaf Een wiskundige structuur bestaande uit knopen en verbindingen, gericht of ongericht. 45

grafentheorie De wiskundige studie van grafen, bestaande uit knopen en verbindingen, met toepassingen in netwerken, routes en structuren. 45

groepentheorie De studie van algebraïsche structuren genaamd groepen, gebruikt in meetkunde en symmetrieanalyse. 34

hoekpunt Een punt waar twee of meer randen samenkomen, bijvoorbeeld in een veelvlak of een graaf. 19, 34

inductie Een wiskundige bewijsstrategie waarbij een stelling voor alle natuurlijke getallen wordt bewezen door een basisgeval te controleren en een inductiestap toe te passen. 59

irrationeel getal Een getal dat niet als een breuk van twee gehele getallen kan worden geschreven, zoals π of $\sqrt{2}$. 20

knoop Een punt in een graaf dat een object of locatie vertegenwoordigt en verbonden kan zijn met andere knopen. 45

lijnsegment Een rechte lijn begrensd door twee eindpunten. 32, 54

lijnstuk Synoniem voor lijnsegment. 32

lineaire ongelijkheid Een vergelijking waarin lineaire uitdrukkingen worden beperkt door ongelijkheidsoperatoren zoals $>$ of $\leq$. 32

loodrecht Een meetkundige relatie waarbij twee lijnen, vlakken of een lijn en een vlak een hoek van 90° vormen. 54

matrix Een rechthoekige tabel van getallen of symbolen die lineaire transformaties kan representeren. 32

meetkunde De wiskundige discipline die vormen, ruimten en de eigenschappen daarvan bestudeert, zoals afstanden, hoeken en oppervlakken. 13

minimal spanning tree (MST) Een deelverzameling van een graaf die alle knopen verbindt met de minimale som van gewichten, zonder cycli. 51

netstructuur Een $(n-1)$-dimensionale uitvouwing van een n-dimensionale polytoop die alle facetten en hun verbindingen toont. 13

omhulling De kleinste convexe vorm die een verzameling punten volledig omvat. 31

ongerichte graaf Een graaf waarbij verbindingen geen richting hebben en eenvoudig knopen met elkaar verbinden. 46

ongewogen graaf Een graaf waarbij verbindingen geen specifieke waarden of gewichten hebben. 47

ontwikkeling Het proces van uitvouwen van een polytoop naar een netstructuur. 13, 39

perfecte matchingsgraaf Een graaf waarin elke knoop exact één andere knoop is verbonden, zodat iedere knoop deel uitmaakt van een perfecte matching. 89

polyhedron Een 3D-geometrisch object met vlakke facetten, rechte randen en hoekpunten, zoals een kubus of een piramide (veelvlak). 14, 31

polytoop Een geometrisch object dat zich in n dimensies uitstrekt met $n \geq 0$, zoals een driehoek (2D), een piramide (3D), of een tesseract (4D). 13

projectie De weergave van een hoger-dimensionaal object op een lager-dimensionale ruimte, bijvoorbeeld een schaduw of een perspectieftekening. 27

recursie Een methode waarbij een functie of definitie zichzelf herhaalt om een probleem op te lossen door het op te splitsen in kleinere subproblemen. 56

regelmatige platonische lichamen Veelvlakken met gelijke zijden, randen en hoeken, zoals een tetraëder, kubus of dodecaëder. 43

relativiteitstheorie Een natuurkundige theorie die de eigenschappen van ruimtetijd beschrijft, ontwikkeld door Einstein, met speciale en algemene relativiteit. 19

ribbe Een rechte rand van een veelvlak of polytoop. 39

Riemann-ruimte Een gebogen meetkundige ruimte waarin de eigenschappen van ruimtetijd kunnen worden beschreven, zoals in de algemene relativiteitstheorie. 19

simplex Een generalisatie van een driehoek naar meer dimensies, bijvoorbeeld een tetraëder (3D) of een 4-simplex (4D). 32

spanning tree Een deelverzameling van een graaf die alle knopen verbindt zonder cycli te vormen. 50

tesseract Een 4D-polytoop, de vierdimensionale analoog van een kubus. 32

tetraëder Een 3D-vorm met vier driehoekige zijden, een simplex in drie dimensies. 31

topologie De wiskundige studie van vormen en ruimten die niet veranderen onder continue vervormingen. 40

transformatie Een bewerking zoals roteren, schalen of verschuiven die een geometrisch object verandert zonder de basisstructuur aan te tasten. 20

vector Een wiskundig object met zowel grootte als richting, gebruikt in de meetkunde en natuurkunde. 22

veelvlak Een 3D-geometrisch object met vlakke zijden, randen en hoekpunten, zoals een piramide of een prisma. 31

verbinding Een lijn of rand tussen twee knopen in een graaf die hun relatie of connectie representeert. 45

vertex Een hoekpunt van een geometrisch object zoals een polytoop of een graaf. 19, 32

vlak Een tweedimensionale, oneindig uitgestrekte geometrische structuur. 32

zwaartepunt Het punt in een object of systeem waar de massa of het gewicht als geconcentreerd kan worden beschouwd, vaak berekend als het gemiddelde van de posities van alle massa-elementen, gewogen naar hun massa. 76

Bibliografie

[1] @tesseralis. *Polyhedra Viewer*. Online. Geraadpleegd: 2024-12-04. URL: https://polyhedra.tessera.li/.

[2] 3Blue1Brown. *Why 4D Geometry Makes Me Sad*. Geraadpleegd: 2024-12-04. 2024. URL: https://www.youtube.com/watch?v=piJkuavhV50.

[3] E. Abbott. *Flatland*. Penguin Science Fiction. Penguin Books Limited, 2020. ISBN: 9780141992891.

[4] Gijs Korthals Altes. *Polyhedra.net*. Online. Geraadpleegd: 2024-12-04. URL: https://www.polyhedra.net/nl/.

[5] José Maria Filardo Bassalo, Francisco Caruso en Vitor Oguri. „The fourth dimension: from its spatial nature in Euclidean geometry to a time-like component of non-Euclidean manifolds". In: *Revista Brasileira de Ensino de Física* 43 (2021), e20210034.

[6] J. A. Bondy en U. S. R. Murty. *Graph Theory*. Springer, 2008.

[7] Marc ten Bosch. *Miegakure [Hide and Reveal]*. Geraadpleegd: 2024-12-04. URL: https://miegakure.com/.

[8] Polytope Wiki Community. *Polytope Wiki*. Geraadpleegd: 2024-12-04. 2024. URL: https://polytope.miraheze.org.

[9] H.S.M. Coxeter. *Introduction to Geometry*. Wiley Classics Library. Wiley, 1989. ISBN: 9780471504580.

[10] H.S.M. Coxeter. *Regular Complex Polytopes*. Cambridge University Press, 1991. ISBN: 9780521394901.

[11] H.S.M. Coxeter. *Regular Polytopes*. Dover books on advanced mathematics. Dover Publications, 1973. ISBN: 9780486614809.

[12] G. Cunningham, M. Mixer en E. Schulte. *Polytopes and Discrete Geometry*. Contemporary Mathematics. American Mathematical Society, 2021. ISBN: 9781470448974.

[13] E.D. Demaine en J. O'Rourke. *Geometric Folding Algorithms: Linkages, Origami, Polyhedra*. Children's Book Council of Australia awards 2009. Cambridge University Press, 2007. ISBN: 9780521857574.

[14] Therese Biedly Erik Demainezx Martin Demainez e.a. „Unfolding some classes of orthogonal polyhedra". In: (1998).

[15] René Descartes. *La géométrie*. fr. 1991.

[16] Rene Descartes. *Discours de la Méthode*. fr. Paris, France: Librairie generale francaise, jan 1978.

[17] Satyan L. Devadoss, Matthew S. Harvey en Sam Zhang. „Visualizing and Unfolding Nets of 4-Polytopes". In: *38th International Symposium on Computational Geometry (SoCG 2022)*. Deel 224. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany, 2022, 67:1–67:4.

[18] Albrecht Dürer. *The Painter's Manual*. Originele titel: *Underweysung der Messung*. N/A, 1525.

[19] Euclides. *The Thirteen Books of Euclid's Elements*. Red. door Thomas L. Heath. New York: Dover Publications, 1956.

[20] Estela A Gavosto en James R Miller. „Visualization of data on unfolded hypercubes". In: *Journal of Visualization* 16 (2013), p. 85–94.

[21] M. Grinberg. *Polytopes, Graphs, and Complexes*. Springer, 2006.

[22] Aric A. Hagberg, Daniel A. Schult en Pieter J. Swart. *NetworkX*. `https://networkx.org/documentation/stable/`. Geraadpleegd: 2024-12-04. 2008.

[23] Junior Einstein. *Bouwplaten - Ruimtelijk Inzicht - Wiskunde Leerjaar 6*. Geraadpleegd: 2024-11-24. 2024. URL: `https://wiskunde.junioreinstein.be/wiskunde-leerjaar-6/meetkunde/ruimtelijk-inzicht/bouwplaten`.

[24] Sandeep Koranne en Anand Kulkarni. „Combinatorial Polytope Enumeration". In: (aug 2009).

[25] Terence Lynch. „The Geometric Body in Dürer's Engraving Melencolia I". In: *Journal of the Warburg and Courtauld Institutes* 45 (1982), p. 226–232. ISSN: 00754390. URL: `http://www.jstor.org/stable/750979` (bezocht op 19-08-2024).

[26] HyperCubist Math. *Visualizing 4D pt 2: The Stack Game*. Geraadpleegd: 2024-12-04. 2024. URL: `https://www.youtube.com/watch?v=ZmRK9J3GkhM`.

[27] HyperCubist Math. *Visualizing 4D Pt.1*. Geraadpleegd: 2024-12-04. 2024. URL: `https://www.youtube.com/watch?v=SwGbHsBAcZ0`.

[28] Stand-up Maths. *Can the Same Net Fold into Two Shapes?* Geraadpleegd: 2024-12-01. 2022. URL: https://www.youtube.com/watch?v=jOTTZtVPrgo.

[29] Jack Murtagh. „How the Seven Bridges of Königsberg Spawned New Math". In: *Scientific American* (apr 2024). Geraadpleegd: 2024-11-11.

[30] mwalczyk. *four*. GitHub. Geraadpleegd: 2024-12-04. 2019. URL: https://github.com/mwalczyk/four.

[31] Numberphile. *Perfect Shapes in Higher Dimensions*. Geraadpleegd: 2024-12-04. 2016. URL: https://www.youtube.com/watch?v=2s4TqVAbfz4.

[32] OEIS Foundation Inc. *The On-Line Encyclopedia of Integer Sequences*. Published electronically at http://oeis.org. 2024.

[33] Matt Parker. *How many 3D nets does a 4D hypercube have?* 2021. URL: https://www.youtube.com/watch?v=Yq3P-LhlcQo.

[34] Keith Paton. „An algorithm for finding a fundamental set of cycles of a graph". In: *Communications of the ACM* 12.9 (1969), p. 514–518.

[35] Tune H Pers e.a. „The validation and assessment of machine learning: a game of prediction from high-dimensional data". In: *PLoS One* 4.8 (2009), e6287.

[36] Trun Ramteke. „Unfoldings of 4D-Hypercube Unfoldings that tile R2". In: sep 2022.

[37] SharkD. *Various projections of cube above plane*. https://commons.wikimedia.org/wiki/File:Various_projections_of_cube_above_plane.svg. 2016.

[38] Geoffrey C Shephard. „Convex polytopes with convex nets". In: *Mathematical Proceedings of the Cambridge Philosophical Society*. Deel 78. 3. Cambridge University Press. 1975, p. 389–403.

[39] Raphael Straub en Hartmut Prautzsch. *Creating optimized cut-out sheets for paper models from meshes*. Citeseer, 2011.

[40] Max Tegmark. „On the dimensionality of spacetime". In: *Classical and Quantum Gravity* 14.4 (1997), p. L69.

[41] Neil deGrasse Tyson en Robert Krulwich. *A mind-expanding tour of the cosmos with Neil deGrasse Tyson and Robert Krulwich*. YouTube. 2017. URL: https://www.youtube.com/watch?v=AyAK3QBnMGQ.

[42] State Library Victoria. *Books that changed the world*. Accessed: 2025-02-03. 2018. URL: https://blogs.slv.vic.gov.au/arts/books-that-changed-the-world/.

[43] Lars Zawallich. „Unfolding polyhedra via tabu search". In: *The Visual Computer* (2024), p. 1–14.

Bijlage A

Afstand tussen punten en het inproduct

In de Euclidische ruimte is de afstand tussen twee punten een belangrijke eigenschap. Deze afstand is nauw verwant aan het inproduct van twee vectoren. Stel je voor dat we twee punten $x = (x_1, x_2, \ldots, x_n)$ en $y = (y_1, y_2, \ldots, y_n)$ hebben in een ruimte met n dimensies (zoals $\mathbb{R}^n$). De afstand d tussen deze twee punten wordt gedefinieerd door de volgende formule:

$$d = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \cdots + (x_n - y_n)^2} \tag{A.1}$$

Deze formule geeft ons de 'rechte lijn' afstand tussen de twee punten. Nu gaan we deze formule afleiden met behulp van het inproduct van twee vectoren.

Het inproduct (ook wel scalair product genoemd) van twee vectoren $\boldsymbol{x}$ en $\boldsymbol{y}$ wordt als volgt berekend:

$$\boldsymbol{x} \cdot \boldsymbol{y} = x_1 y_1 + x_2 y_2 + \cdots + x_n y_n$$

Nu willen we de afstand tussen de twee punten x en y vinden door te kijken naar het verschil tussen de twee vectoren, dus we nemen de vector $\boldsymbol{x} - \boldsymbol{y}$. Het inproduct van deze verschilvector met zichzelf is:

$$(\boldsymbol{x} - \boldsymbol{y}) \cdot (\boldsymbol{x} - \boldsymbol{y}) = (x_1 - y_1)^2 + (x_2 - y_2)^2 + \cdots + (x_n - y_n)^2$$

Je ziet dat dit precies hetzelfde is als de formule voor de afstand die we eerder zagen, maar dan zonder de vierkantswortel. Dit betekent dat we de afstand d tussen de vectoren $\boldsymbol{x}$ en $\boldsymbol{y}$ kunnen schrijven als:

$$d = \sqrt{(\boldsymbol{x} - \boldsymbol{y}) \cdot (\boldsymbol{x} - \boldsymbol{y})}$$

Deze afleiding laat zien dat de Euclidische afstand tussen twee punten heel een-

voudig berekend kan worden met het inproduct van de verschilvector tussen die twee punten. Het voordeel hiervan is dat je de afstand kunt berekenen zonder expliciet elke afzonderlijke coördinaat te moeten vergelijken. In plaats daarvan gebruik je de kracht van het inproduct, wat de berekening makkelijker maakt. Dit wordt dus ook regelmatig gebruikt in software, om berekeningen te versnellen.

Bijlage B

Dimensionaliteit van een polytoop berekenen

B.1 Algebraïsch

De dimensionaliteit van een polytoop in een n-dimensionale ruimte geeft aan in welke affiene deelruimte de polytoop ligt. Hieronder leggen we stap voor stap uit hoe de dimensionaliteit wordt berekend. Dit proces is ook uitgewerkt in Python 3, te vinden in bijlage B.2.

Stap 1: Definieer de hoekpunten van de polytoop

Een polytoop P wordt beschreven door een verzameling hoekpunten:

$$\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_m, \quad \text{waarbij } \boldsymbol{v}_i \in \mathbb{R}^n.$$

Elk hoekpunt $\boldsymbol{v}_i$ is een vector in een n-dimensionale ruimte. Samen vormen deze punten de geometrische structuur van de polytoop.

Stap 2: Vind de affiene deelruimte

De affiene deelruimte waarin de polytoop ligt, is de kleinste ruimte die alle hoekpunten bevat. Om deze ruimte te vinden, gebruiken we het *affiene span* van de hoekpunten. Het affiene span is een concept uit de lineaire algebra dat ons helpt de kleinste affiene ruimte te vinden die een gegeven set punten bevat.

Formeel wordt het affiene span van de hoekpunten $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_m$ gedefinieerd als:

$$\text{Aff}(\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_m) = \left\{ \sum_{i=1}^{m} \lambda_i \boldsymbol{v}_i \;\middle|\; \sum_{i=1}^{m} \lambda_i = 1 \text{ en } \lambda_i \in \mathbb{R} \right\}. \tag{B.1}$$

Deze formule betekent dat elk punt in de affiene deelruimte kan worden uitgedrukt als een gewogen som van de hoekpunten. Hierbij zijn de gewichten (λ_i) reële getallen die optellen tot 1.

1. **Gewogen Som van Hoekpunten:** Een punt $\boldsymbol{p}$ in de affiene deelruimte kan worden geschreven als een lineaire combinatie van de hoekpunten, met gewichten $\lambda_1, \lambda_2, \ldots, \lambda_m$. Bijvoorbeeld, als je drie hoekpunten hebt $(\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3)$, dan kan elk punt in de affiene deelruimte worden uitgedrukt als $\lambda_1\boldsymbol{v}_1 + \lambda_2\boldsymbol{v}_2 + \lambda_3\boldsymbol{v}_3$.

2. **Som van de Gewichten:** Het kenmerkende aan een affiene combinatie (in tegenstelling tot een gewone lineaire combinatie) is dat de gewichten optellen tot 1. Dit zorgt ervoor dat de affiene deelruimte precies groot genoeg is om alle hoekpunten te bevatten, maar niet groter. Deze voorwaarde ($\sum_{i=1}^{m} \lambda_i = 1$) helpt ons om een unieke beschrijving van de affiene deelruimte te krijgen.

3. **Interpretatie van λ_i:** De waarden van λ_i kunnen zowel positief als negatief zijn, wat betekent dat de affiene deelruimte zowel in als buiten de convexe omhulling van de hoekpunten kan liggen. De convexe omhulling is het kleinste convexe polytoop dat alle hoekpunten bevat, en de affiene deelruimte kan deze convexe omhulling uitbreiden naar een lineaire subruimte.

$$\text{Aff}(\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_m) = \{\lambda_1\boldsymbol{v}_1 + \lambda_2\boldsymbol{v}_2 + \cdots + \lambda_m\boldsymbol{v}_m \mid \lambda_1 + \lambda_2 + \cdots + \lambda_m = 1\}$$

Stap 3: Verschuif de oorsprong

Om de berekening eenvoudiger te maken, kiezen we één hoekpunt als referentiepunt. We nemen bijvoorbeeld $\boldsymbol{v}_1$ en herschrijven de andere hoekpunten relatief ten opzichte van dit punt. Dit geeft ons een nieuwe set vectoren:

$$\boldsymbol{w}_i = \boldsymbol{v}_i - \boldsymbol{v}_1 \quad \text{voor alle } i \in \{2, 3, \ldots, m\}.$$

Deze vectoren $\boldsymbol{w}_i$ beschrijven de relatieve positie van de overige hoekpunten ten opzichte van $\boldsymbol{v}_1$. Dit maakt de berekening van de dimensie eenvoudiger.

Stap 4: Vorm de matrix van relatieve vectoren

Vervolgens vormen we een matrix W waarin de vectoren $\boldsymbol{w}_i$ kolommen zijn:

$$W = \begin{pmatrix} | & | & \cdots & | \\ \boldsymbol{w}_2 & \boldsymbol{w}_3 & \cdots & \boldsymbol{w}_m \\ | & | & \cdots & | \end{pmatrix}.$$

Elke kolom van W komt overeen met een vector $\boldsymbol{w}_i$. De dimensie van de affiene deelruimte wordt bepaald door te kijken hoeveel van deze vectoren onafhankelijk van elkaar zijn.

Stap 5: Bereken de rang van de matrix

De rang van W wordt berekend. De rang is het aantal lineair onafhankelijke kolommen in de matrix. Lineair onafhankelijke vectoren beschrijven de dimensie van de kleinste lineaire ruimte waarin deze vectoren liggen.

Stap 6: Bereken de dimensie van de affiene deelruimte

De dimensie van de affiene deelruimte waarin de polytoop ligt, wordt gegeven door:

$$\text{Dim(Affiene Deelruimte)} = \text{Rang}(W) + 1. \tag{B.2}$$

De $+1$ wordt toegevoegd omdat we eerder één hoekpunt ($\boldsymbol{v}_1$) als referentiepunt hebben genomen, wat een dimensie toevoegt aan de affiene ruimte. Bijvoorbeeld, als de rang van W gelijk is aan k, dan ligt de polytoop in een affiene deelruimte van dimensie $k + 1$.

Stap 7: Controleer de dimensie

Als de dimensie van de affiene deelruimte kleiner is dan n, betekent dit dat de polytoop in een lagere-dimensionale subruimte van de n-dimensionale ruimte ligt. Als de dimensie gelijk is aan n, ligt de polytoop in de volledige n-dimensionale ruimte.

B.2 In Python

```
import numpy as np

def affiene_hull(points):
    points = np.array(points)

    if points.shape[0] < 2:
        raise ValueError("Ten minste 2 punten nodig om
            ↪ affiene deelruimte te vinden.")

    vectors = points - points[0]
    rank = np.linalg.matrix_rank(vectors)
    dimension = rank

    return points, dimension

# Voorbeeld: vierkant in 3D
if __name__ == "__main__":
    points = np.array([
        [1, 2, 4],
        [2, 3, 5],
        [3, 2, 2],
        [4, 5, 7]
    ])

    hull_points, hull_dimension = affiene_hull(points)

    print("Dimensie van de affiene deelruimte:",
        ↪ hull_dimension)
```

https://gist.github.com/BramLeisink/7881a2d553800bbd061bee2af00b55e6

Bijlage C

Suggesties voor verder onderzoek

In deze bijlage lichten we de vijf onderzoekvragen die we in de discussie hebben besproken verder toe.

C.1 Is elke polytoop te definiëren door slechts hoekpunten en hun relaties?

Een interessante vraag is of elke n-dimensionale polytoop volledig gedefinieerd kan worden door:

1. Een verzameling hoekpunten, elk voorgesteld door hun coördinaten in de n-dimensionale ruimte;

2. Een verzameling paren van hoekpunten, die samen de randen van de polytoop beschrijven.

Dit idee komt voort uit het concept dat een polytoop te beschouwen is als een verzameling facetten, opgebouwd uit objecten van lagere dimensies, met als fundament de relaties tussen hoekpunten en randen. We verwachten dat door deze structuur recursief te doorlopen, de unieke polytoop die overeenkomt met deze gegevens volledig is vastgelegd.

Voorlopige hypothese

We stellen daarom de volgende hypothese:

> *Elke n-dimensionale polytoop kan volledig worden gedefinieerd door een eindige verzameling hoekpunten (coördinaten in n-ruimte) en een eindige verzameling van hun onderlinge verbindingen (paren van hoekpunten die randen vormen).*

Onderzoek en uitdaging

Hoewel de bovenstaande hypothese intuïtief correct lijkt, zijn er uitdagingen bij het bewijs ervan:

- **Uniekheid**: Het is niet bewezen dat deze methode altijd leidt tot een unieke polytoop. Kunnen meerdere verschillende polytopen dezelfde verzameling hoekpunten en verbindingen delen?
- **Complexiteit**: De structuur van een polytoop is veel complexer dan alleen hoekpunten en randen. Polytopen hebben facetten van hogere dimensies die mogelijk niet volledig en uniek kunnen worden vastgelegd met alleen een lijst van paren.
- **Verlies van informatie**: De verzameling hoekpunten en verbindingen bevat mogelijk niet genoeg informatie over hoe de facetten of hogere-dimensionale structuren moeten worden opgebouwd.

Een mogelijke aanpak om deze vragen te onderzoeken is door het algoritme verder te ontwikkelen en te testen op bekende voorbeelden van polytopen.

Vooruitzicht

Het idee dat elke polytoop uniek gedefinieerd kan worden door een combinatie van hoekpunten en hun verbindingen zou grote implicaties hebben voor de representatie en het manipuleren van polytopen. Dit zou immers betekenen dat:

- Er een zeer compacte en efficiënte datastructuur voor polytopen mogelijk is;
- Geometrische problemen over polytopen kunnen worden herleid tot puur combinatorische vraagstukken.

Voorlopig blijven deze ideeën speculatief en zonder sluitend bewijs. Verdere mathematische en computationele experimenten zijn nodig om deze hypothese te testen en eventuele tegenvoorbeelden te vinden. We hebben zelf veel tijd aan deze vraag besteed, maar zijn helaas niet tot een mooie conclusie gekomen.

C.2 Is er een functie $f(n)$ voor het aantal unieke netstructuren van een n-kubus?

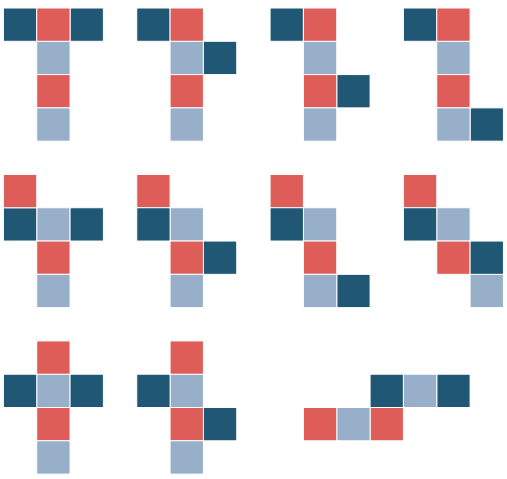
n	$a(n)$
2	1
3	11
4	261
5	9 694
6	502 110
7	33 064 966
8	2 642 657 228
9	248 639 631 948
10	26 941 775 019 280

Tabel C.1: Het aantal unieke netten voor een n-kubus [32, A091159]

De bovenstaande tabel geeft het aantal unieke netten van een n-kubus weer. Hoe zijn deze getallen gevonden? Gewoon door te proberen. Een interessante vraag die hierdoor ontstaat is: Is er een manier om deze getallen te berekenen of is er een functie $f(n)$ die het aantal unieke netten van een n-kubus beschrijft?

Wij hopen dat door de eigenschappen van de facet-adjacentiegraaf van de n-kubus beter te bestuderen dit vraagstuk op te lossen is. De eigenschappen die wij hebben omschreven kunnen een begin vormen, zoals dat dit een complementaire graaf van een perfecte matchingsgraaf is.

Een uitstekende introductie tot dit onderwerp is de video van Matt Parker hierover: `https://www.youtube.com/watch?v=Yq3P-LhlcQo` [33]



Figuur C.1: Alle 11 netstructuren van een 3-kubus

C.3 Is elke convexe n-dimensionale polytoop te ontwikkelen naar een netstructuur zonder overlap?

Deze vraag speelt een centrale rol in ons onderzoek en is een natuurlijke generalisatie van de vraag voor 3-dimensionale polytopen. Voor 3D-polytopen is deze vraag al beantwoord. De volgende stelling, beter bekend als de **Stelling van Alexandrov**, geeft een duidelijk antwoord voor convexe 3D-polytopen:

Stelling C.3.1 (Stelling van Alexandrov)**.** *Elke convexe 3-dimensionale polytoop heeft ten minste één niet-overlappende netstructuur.*

Met andere woorden: je kunt elke convexe polytoop in drie dimensies uitvouwen tot een vlak figuur waarbij de vlakken elkaar niet overlappen. Maar zelfs in drie dimensies roept dit probleem al interessante vragen op. De stelling zegt bijvoorbeeld niet dat er maar één manier is om dit te doen, en het geeft ook geen handige methode om zo'n netstructuur te construeren.

De grote uitdaging zit nu in het doortrekken van deze stelling naar hogere dimensies. Zodra $n \geq 4$ wordt, krijgen we te maken met veel complexere structuren. Daarnaast groeit het aantal mogelijke manieren om een polytoop uit te vouwen enorm naarmate de dimensies toenemen.

Hoewel er nog geen algemeen bewijs is dat elke convexe n-dimensionale polytoop een niet-overlappende netstructuur heeft, weten we al wel een paar dingen:

- Voor bepaalde soorten polytopen, zoals simplexen, is bewezen dat ze altijd een niet-overlappende netstructuur hebben.

- Uit onderzoek van Shephard [38] blijkt dat dit probleem sterk afhankelijk is van de specifieke meetkundige en combinatorische eigenschappen van het polytoop.

- Meer recent werk, zoals dat van Devadoss en O'Rourke [17], richt zich op het gebruik van algoritmes en computerberekeningen om deze structuren te vinden.

Dit alles laat zien dat er nog heel wat te ontdekken valt als het gaat om netstructuren in hogere dimensies.

C.4 Zijn er naast de tetraëder, andere polytopen die een netstructuur kunnen hebben met een convexe buitenste rand?

Deze vraag gaat een stapje verder dan alleen het uitvouwen van polytopen. Het draait namelijk om een specifieke eigenschap van de netstructuur: kan de buitenste rand van het net altijd convex zijn? Bij een convex object buigen alle hoeken naar buiten.

Voor de tetraëder is dit vrij eenvoudig te zien. Als je een tetraëder uitvouwt, kun je een netstructuur maken waarin de buitenste rand altijd convex is. Maar zodra je naar complexere polytopen kijkt - zoals een kubus, een dodecaëder of zelfs hogere-dimensionale objecten - is dit allesbehalve vanzelfsprekend.

De vraag wordt dan: zijn er andere polytopen, naast de tetraëder, waarvoor dit altijd lukt? En zo niet, wat maakt de tetraëder uniek in dit opzicht?

C.5 Wat zijn heuristieken om overlap in hoger-dimensionale netstructuren te voorkomen?

Door middel van heuristieken is overlap in netstructuren te voorkomen. Door middel van heuristieken kan aangeven worden hoe handig of niet-handig het is om twee facetten aan elkaar te laten zitten. Door deze heuristieken slim te ontwerpen, is het zo mogelijk om, zonder al te veel computerkracht, een netstructuur zonder overlap te vinden. Dit is des te belangrijker bij het toepassen van ons algoritme bij polytopen van een hoge dimensie, van wege de grote hoeveelheid computaties die deze vereisen.

In *Creating optimized cut-out sheets for paper models from meshes* worden verschillende heuristieken voor het ontwikkelen naar twee dimensies gepresenteerd. Voor het ontwikkelen van polytopen met een hogere dimensie, zijn echter nog geen heuristieken ontworpen. We hebben het hier al eerder over gehad in paragraaf 9.3.1. Het ontwerpen van deze heuristiken kan een grote verbetering zijn voor onze methode, omdat het de hoeveelheid overlap drastisch verminderd.

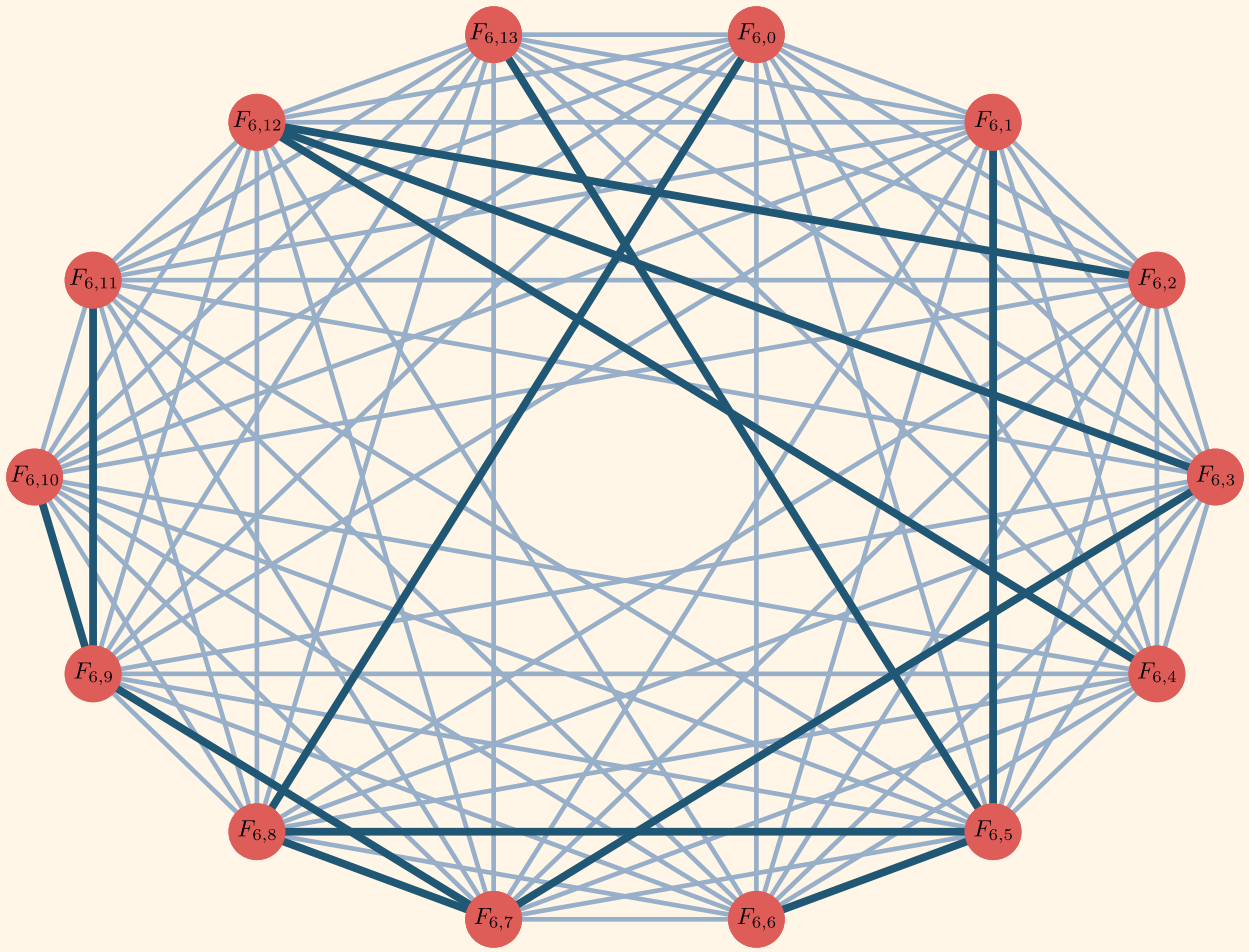
Bijlage D

Gerelateerde materialen

In dit hoofdstuk hebben we een verzameling artikelen, video's en andere bronnen verzameld die losjes gerelateerd zijn aan het onderwerp van dit profielwerkstuk. Deze materialen bieden een breder perspectief op de wiskundige concepten die zijn behandeld, en kunnen je inspireren om dieper in te gaan op bepaalde onderwerpen of ze vanuit een ander oogpunt te bekijken.

Hieronder vindt je een lijst van interessante artikelen, websites en video's die we hebben geselecteerd:

- How many 3D nets does a 4D hypercube have? - Stand-up Maths - YouTube [33]
- Why 4D Geometry Makes Me Sad - 3Blue1Brown - YouTube [2]
- Visualizing 4D Pt.1 - HyperCubist Math - YouTube [27]
- Visualizing 4D Pt.2: Visualizing 4D pt 2: The Stack Game - HyperCubist Math - YouTube [26]
- Perfect Shapes in Higher Dimensions - Numberphile - YouTube [31]
- Can the Same Net Fold into Two Shapes? - Stand-up Maths - YouTube [28]
- polyhedra.net - Gijs Korthals Altes - Website [4]
- Polyhedra Viewer - @tesseralis - Website [1]
- four - Michael Walczyk - Code (rs) [30]
- Miegakure [Hide and Reveal] - Game [7]
- Polytope Wiki - Website [8]


$F_{6,13}$
$F_{6,0}$
$F_{6,12}$
$F_{6,1}$
$F_{6,11}$
$F_{6,2}$
$F_{6,10}$
$F_{6,3}$
$F_{6,9}$
$F_{6,4}$
$F_{6,8}$
$F_{6,5}$
$F_{6,7}$
$F_{6,6}$