

BlindsolveDB

Webapplicatie voor het leren en optimaliseren van geavanceerde
Rubik's cube blindsolving methodes

INHOUDSTAFEL

Lijst van afkortingen	3
Lijst van bijlagen	4
Introductie	5
Preproductie	6
Doelstellingen	6
Onderzoek onderwerp	6
Onderzoek doelgroep	11
Onderzoek concurrentie	18
Onderzoeksconclusies	22
Productie	23
Omschrijving	23
Scope	23
Planning	24
Vorbereidingen	24
Uitvoering	28
Besluit	30
Conclusies	30
Reflectie	30
Referentielijst	31
Bijlages	33
Bijlage 1: WCA events en formats	34
Bijlage 2: Speedsolve verloop	35
Bijlage 3: Rubik's cube turns	36
Bijlage 4: Speffz lettering scheme	37
Bijlage 5: Memorisatiemethodes	38

LIJST VAN AFKORTINGEN

3BLD	Het geblinddoekt oplossen van een 3x3x3 Rubik's kubus. Een van de WCA events. 4BLD en 5BLD verwijzen respectievelijk naar de 4x4x4 en 5x5x5 kubus.
3x3	3x3x3 Rubik's kubus. Andere n-dimensionale kubusvormige puzzels worden gelijkaardig afgekort als nxn.
ao5	Average of 5. De gemiddelde tijd van vijf resultaten, waarbij de snelste en traagste tijd weggelaten worden.
AST	Abstract syntax tree
BLD	Verzamelnaam voor alle blindsolving events.
bo3	Best of 3. De snelste tijd van drie resultaten.
CDN	Content Delivery Network
MBLD	Multi-Blind, het geblinddoekt oplossen van zoveel mogelijk 3x3x3 Rubik's kubussen binnen een bepaalde tijdslimiet. Een van de WCA events.
mo3	Mean of 3. De gemiddelde tijd van drie resultaten.
ORM	Object-relational mapping
QOL	Quality of life
VPS	Virtual private server
WCA	World Cubing Association. Een organisatie die speedcubing competities organiseert, records bijhoudt en het reglement bepaalt.

LIJST VAN BIJLAGEN

- Bijlage 1: WCA events en formats
- Bijlage 2: Speedsolve verloop
- Bijlage 3: Rubik's cube turns
- Bijlage 4: Speffz lettering scheme
- Bijlage 5: Memorisatiemethodes

INTRODUCTIE

Voor deze bachelorproef heb ik gekozen voor een onderwerp waar ik zelf vaak mee bezig ben. In 2016 heb ik namelijk geleerd de Rubik's Cube op te lossen, wat leidde tot de ontdekking van een nieuwe hobby: speedcubing. In 2018 deed ik mee aan mijn eerste competitie, waardoor ik interesse kreeg in het geblinddoekt oplossen van de puzzel. Het jaar erop haalde ik Belgisch record in het blindsolving event.

Tijdens het pad dat ik heb afgelegd, heb ik ontdekt dat er weinig goede tools zijn voor leren van blindsolving. Het doel van mijn bachelorproef is een mediaproduct ontwikkelen dat zowel nieuwe als ervaren blindsolvers helpt met het leren en optimaliseren van geavanceerde blindsolving methodes.

Ten eerste wordt in het onderdeel 'Preproductie' een overzicht gegeven van de uitgevoerde onderzoeken: onderwerp, doelgroep en concurrentie. Er wordt dieper ingegaan op wat speedcubing inhoudt en de voornaamste gerelateerde termen worden uitgelegd. Verder maakt de lezer kennis met het concept blindsolving en de begrippen daaromtrent. In het doelgroeponderzoek worden de resultaten van enkele grote enquêtes betreffende speedcubing besproken, met een focus op blindsolving. In de concurrentieanalyse wordt gekeken naar tools die reeds gebruikt worden door blindsolvers en worden de voor- en nadelen afgewogen. Ook worden enkele gekende speedcubing websites die zich niet focussen op blindsolving, geanalyseerd.

Ten tweede komt de uitwerking van het werkstuk aan bod in het onderdeel 'Productie'. Hierin wordt het werkstuk omschreven, de scope uitgelegd en de planning opgesteld. Verder wordt de voorbereiding overlopen, met name de wireframes, het data model en de tech stack. Als laatste wordt er dieper ingegaan op de uitvoering van het werkstuk.

Ten slotte worden conclusies getrokken uit het resultaat en wordt gereflecteerd op het onderzoek en het werkstuk.

PREPRODUCTIE

Doelstellingen

Aan de hand van dit onderzoek wil ik uiteenzetten wat blindsolving precies inhoudt en hoe het aangeleerd wordt. Bijkomend wil ik aantonen wat de struikelblokken zijn in het leerproces en wat speedsolvers ervan weerhoudt om blindsolving te leren. Verder wordt onderzocht wie de doelgroep is en welke tools er reeds bestaan, zodat een applicatie ontwikkeld kan worden die aangepast is aan de doelgroep en tegemoet komt aan de tekortkomingen van de huidige leermethodes.

Onderzoek onderwerp

Onderzoeksvraag

- Wat is speedcubing?
- Wat is blindsolving?
- Waaruit bestaat een blindsolve?
- Welke methodes bestaan er?
- Wat zijn de huidige records?

Methodologie

Desk Research

Resultaten

Speedcubing

Speedcubing, of kortweg cubing, is een sport waarbij combinatiepuzzels, zoals de Rubik's Cube, op een zo snel mogelijke tijd opgelost worden.

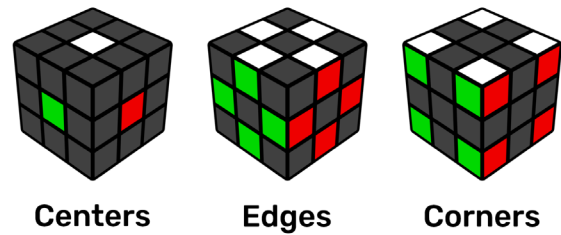
De Rubik's Cube werd uitgevonden in 1974 door Ernő Rubik, en was in 1977 commercieel beschikbaar. In de hierop volgende jaren groeide de puzzel in populariteit en verschenen de eerste competities. In 1982 vond het eerste grootschalige kampioenschap, de World Rubik's Cube Championship (WC1982), plaats. De Rubik's Cube rage stierf echter snel uit sinds 1983 waardoor speedcubing een zogenaamde 'dark age' kende tot het einde van de jaren '90. Door de toenemende populariteit van het internet verschenen verschillende speedcubing websites waarin onder andere tutorials gedeeld werden, met een heropleving van interesse in speedcubing tot gevolg. In 2003 organiseerde Ron van Bruchem World Rubik's Games Championship 2003 (WC2003) in Toronto. Met 89 deelnemers en een hoop media-aandacht, was dit een beduidend succes sinds WC1982. Door het stijgende aantal competities, werd in 2004 het World Cubing Association (WCA) opgericht. Het WCA fungeert als organisatie die competities organiseert, records bijhoudt en het reglement bepaalt. Dit laatste, de standaardisatie van de regels, was een belangrijke mijlpaal binnen het speedcuben. Tot nu toe zijn er al meer dan 7000 competities georganiseerd met meer dan 150.000 deelnemers uit 143 verschillende landen.

In totaal biedt het WCA 17 events aan (zie bijlage 1). Deze zijn gebonden aan officiële regels en worden gehouden op erkende WCA competities. Competities worden georganiseerd door WCA

delegates, die de keuze hebben welke events er zullen plaatsvinden. De delegates overzien tevens dat elke solve op een competitie volgens de regels gebeurt (zie bijlage 2).

Notatie en algoritmes

Een 3x3 bestaat uit drie soorten cubies: centers, edges en corners, die respectievelijk uit één, twee en drie kleurvlakken bestaan. De centers staan vast ten opzichte van elkaar en hun kleur bepaalt dus welke kleur het hele vlak krijgt.



De zes vlakken van een Rubik's Cube of soortgelijke puzzels worden aangeduid met de letters U (*up*), F (*front*), R (*right*), D (*down*), B (*back*) en L (*left*). Eén van deze vlakken draaien (turn) wordt geschreven als de letter van het vlak en slaat op een kwartslag in wijzerzin. Het toevoegen van een enkel aanhalingsteken, geeft een verandering van de richting naar tegenwijzerzin aan. Een halve draai wordt aangeduid door het cijfer 2 achter de letter van het vlak te plaatsen (twee kwartslagen zijn namelijk een halve draai). Zo stelt de letter U een kwartslag in wijzerzin voor van het U-vlak. U' verandert de richting naar tegenwijzerzin. U2 stelt een halve draai van 180° voor. Naast het draaien van een vlak (*face turn*), bestaan er ook *wide turns*, *slice turns* en rotaties (zie bijlage 3). Ook voor deze turns geldt dat een aanhalingsteken of het cijfer 2 toegevoegd kan worden om een andere richting aan te duiden.

Algoritmes zijn, aangaande speedcubing, een opeenvolging van draaien die een gewenst effect hebben op de puzzel. Ze staan toe om specifieke posities van de puzzel op te lossen, zonder de reeds opgeloste delen opnieuw door elkaar te halen. Het resultaat van een algoritme is dat slechts enkele *cubies* - de aparte stukjes van de kubus - verwisseld of gedraaid worden. Algoritmes worden vanbuiten geleerd om een bepaalde positie tijdens een solve zo snel mogelijk op te kunnen lossen.

Een speciale vorm van algoritmes zijn *commutatoren*. Het heeft de vorm A B A' B' (geschreven als [A, B]), waarbij A en B een sequentie van draaien voorstellen, en A' en B' de inversen van deze sequenties. Door A en B uit te voeren gevolgd door hun inversen, heeft een commutator geen effect op een groot deel van de kubus, wat ervoor zorgt dat specifieke delen met elkaar verwisseld kunnen worden. Meer specifiek komt dit erop neer dat exact drie cubies met elkaar verwisseld worden.

Conjugaties lijken op commutatoren en hebben de vorm A B A' [geschreven als [A: B]]. Het verschil met commutatoren is dat B' hier niet wordt uitgevoerd. De sequentie A kan hier aanzien worden als een setup naar een nieuwe positie die door B opgelost wordt, waarna de setup ongedaan gemaakt wordt. Wanneer men bijvoorbeeld een U-draai verwijderd is van een gekend algoritme A, kan de conjugatie U A U' (ofwel [U: A]) uitgevoerd worden.

Zowel commutatoren als conjugaties zijn algoritmes die intuïtief te begrijpen zijn. Eens men weet hoe ze werken, kunnen ze gebruikt worden om eender welke positie op te lossen. Deze typen algoritmes worden uitvoerig gebruikt in de 3-style methode (zie verder).

Blindsolving

Het geblinddoekt oplossen van een Rubik's Cube of afgeleide puzzels heet blindsolving. De geblinddoekte WCA events zijn 3BLD, 4BLD, 5BLD en MBLD, waarbij 3BLD de populairste is. Verschillend van de gewone events, wordt bij een blindsolve de timer gestart terwijl de puzzel nog afgedekt is. De deelnemer neemt zelf de puzzel van onder de cover en begint de memorisatiefase. Na het memoriseren wordt de blinddoek opgezet en mag deze niet meer afgezet worden tot de puzzel losgelaten wordt.

Bij MBLD (3x3x3 Multi-Blind) wordt een zo groot mogelijk aantal van 3x3 kubussen in één keer geblinddoekt opgelost. De deelnemer kiest het aantal puzzels en krijgt een tijdslimiet van n keer 10 minuten, waarbij n het aantal puzzels is, met een maximum van één uur.

Een blindsolve bestaat uit twee fasen, namelijk memorisatie (*memo*) en uitvoering (*execution*). De methodes bij blindsolving verschillen sterk van andere methodes. Bij de gewone 3x3 events worden vaak de CFOP of Roux methode gebruikt, waarin intuïtieve *block building* stappen gecombineerd worden met sets van algoritmes. Omdat het met een blinddoek niet mogelijk is de staat van een kubus mentaal bij te houden na elke draai, worden via een combinatie van specifieke algoritmes en methodes slechts enkele cubies tegelijk met elkaar verwisseld, zonder de rest van de puzzel te beïnvloeden. Met deze techniek is het mogelijk om een 3x3 te memoriseren door slechts een reeks van gemiddeld 20 letters te onthouden.

Memorisatie

De memo fase start met *tracing*. Hierbij wordt vertrokken van een vaste cubie, de *buffer*, waarvoor men de positie zoekt waar het naartoe moet om opgelost te zijn (door te kijken naar de kleuren van de centers). Voor de cubie die zich op deze positie bevindt, wordt dit proces herhaald en zo worden de opeenvolgende cubies gevolgd tot de huidige positie van de buffer opnieuw bereikt wordt. Een voorbeeld om dit te verduidelijken: stel dat er vertrokken wordt van de corner in de UFR (up-front-right) positie en deze corner hoort in de FDL (front-down-left) positie te zijn. De corner die zich in FDL bevindt, moet op zijn beurt naar BUR (back-up-right). Door zo telkens de locatie te traceren voor elke cubie, wordt een reeks verkregen zoals bijvoorbeeld UFR > FDL > BUR > DBL > LUB > RDB > RDF > UFR.

Om de opeenvolgende locaties te memoriseren, kent men elke locatie op de kubus een letter toe, zodat tijdens de tracing een reeks letters gevormd wordt. Het toekennen van letters aan locaties op de kubus heet een *lettering scheme*. Hoewel dit volledig arbitrair te kiezen is, werd in 2010 het Speffz lettering scheme (zie bijlage 4) voorgesteld als standaard met als doel communicatie tussen blindsolvers te vereenvoudigen. Voorgaand voorbeeld zou geschreven kunnen worden als LQXEOP volgens het Speffz lettering scheme. Merk op hoe er twee letters minder zijn dan het aantal locaties verkregen tijdens de tracing. Dit komt omdat de tracing altijd start en eindigt bij de buffer (hier UFR), wat overbodige informatie is om te onthouden. Dus, deze letterreeks is voldoende om tijdens de oplosfase te weten welke cubies verwisseld moeten worden, door de letters terug te koppelen aan de locaties waar ze naar verwijzen. Bij een 3x3 wordt de tracing herhaald voor zowel de corner pieces als de edge pieces en worden dus twee reeksen van letters onthouden. Het gebruiken van een vaste oriëntatie zorgt ervoor dat de tracing sneller kan gebeuren zonder te moeten zoeken welke kleur waar zit. Met wit bovenaan en groen vooraan duurt het bijvoorbeeld niet lang om te weten dat blauw achteraan zit en geel onderaan. Ook een vaste lettering scheme spaart tijd uit tijdens een solve omdat er niet bewust nagedacht moet worden welke letter naar waar verwijst.

Voor het memoriseren van de reeks letters, ook de memo genoemd, bestaan verschillende technieken. Sommige van deze technieken zijn ontleend uit de memory sports, terwijl anderen specifiek gebruikt worden voor blindsolving. De meest voorkomende methodes zijn het gebruik van verhalen, audio, visueel en loci (zie bijlage 5). Voor elke methode wordt vaak chunking toegepast, wat wil zeggen dat de memo onderverdeeld wordt in kleinere stukken, vaak per twee letters (een letterpaar) om het memoriseren te vereenvoudigen.

Uitvoering

Tijdens de memorisatiefase werden een aantal reeksen van letters verkregen die aanduiden waar elke cubie naartoe moet (bv. LQXEOP voor de corners en HQXVNDJBKFM voor edges). Om de cubies terug naar hun opgeloste positie te brengen (bv. L naar Q, Q naar X, ... voor de corners), worden slechts enkele cubies per keer verwisseld, zodat de posities van de niet-opgeloste cubies hetzelfde blijven en dus de memo blijft kloppen. Enkele bekende methodes voor deze uitvoering zijn OP, M2 en 3-style. Voor het oplossen van corners en edges hoeven niet dezelfde methodes gebruikt te worden. Dit wordt dan weergegeven als bijvoorbeeld OP/M2, waarbij OP de corners methode is en M2 de edges methode.

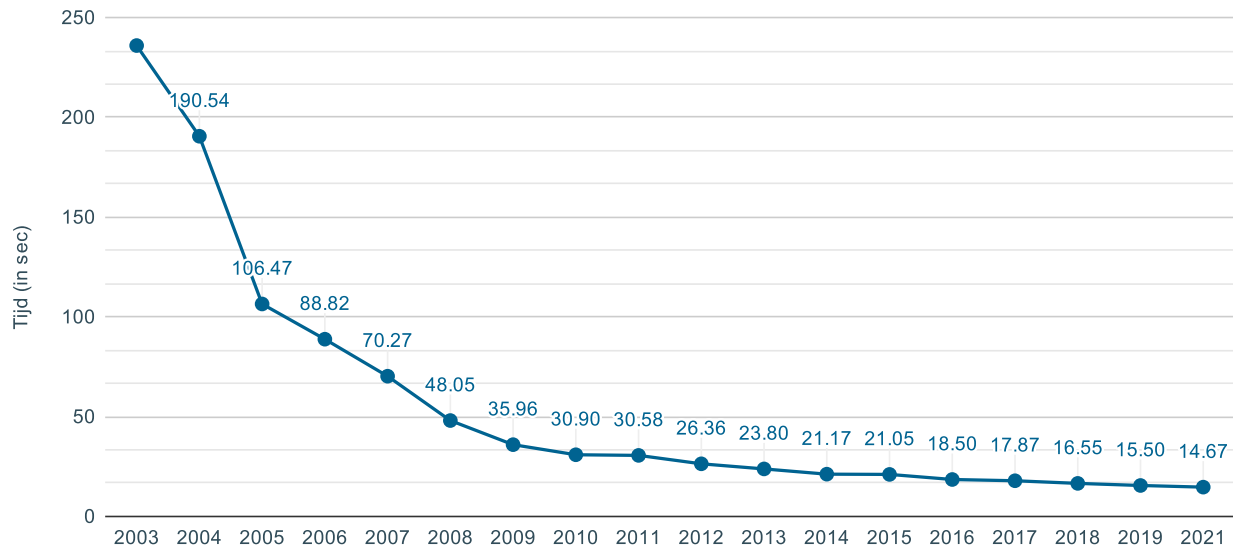
De Old Pochmann (OP) methode is een beginnersmethode die één cubie per keer oplost door middel van één algoritme dat de buffer verwisselt met de cubie in de target positie. Omdat het gebruikte algoritme vrij lang is, is OP geen geschikte methode om sneller te worden.

M2 is een gevorderde methode voor het oplossen van edges en is vernoemd naar het algoritme dat gebruikt wordt. M2 is namelijk een slice turn die de middelste laag 180 graden draait. De M2-draai verwisselt (onder andere) de edges in posities UB en DF (de buffer). Door elke edge één voor één naar UB te brengen met behulp van conjugaties, kunnen heel snel twee edges met elkaar verwisseld worden. Net zoals de OP methode lost M2 de edges één voor één op. Voor corners bestaat een gelijkaardige R2 methode, maar deze wordt zelden gebruikt.

3-style is de snelste methode die gebruikt wordt door geavanceerde blindsolvers. Deze methode dankt zijn naam aan het type algoritmes dat gebruikt wordt, commutatoren, die drie cubies in een cyclus wisselen (zie Notatie en algoritmes). De eerste cubie gaat naar de locatie van de tweede, de tweede naar de locatie van de derde, en de derde keert terug naar de eerste. Hiermee worden twee cubies tegelijk opgelost (de eerste cubie is telkens de buffer), waardoor het aantal algoritmes dat nodig is om de puzzel op te lossen halveert in vergelijking met OP en M2. Dit, in combinatie met de lage *move count* eigen aan commutatoren, maakt de 3-style methode enorm efficiënt. Het nadeel van deze methode is het grote aantal algoritmes dat geleerd moet worden. Door twee cubies tegelijk op te lossen, zijn er 818 mogelijke combinaties die van buiten geleerd moeten worden. Ondanks dat een commutator een intuïtief algoritme is, worden ze vaak uit het hoofd geleerd om geen tijd verloren te laten gaan tijdens een solve.

Records

Het huidige record van 3BLD single staat op naam van Tommy Cherry met een tijd van 14,67 seconden (zie bijlage 1 voor uitleg over events en formats). De oplossing bestond uit 9 algoritmes die op 8,34 seconden uitgevoerd werden. Tommy houdt ook het record van 3BLD mo3 met een tijd van 15,24 seconden. De records voor 4BLD en 5BLD (zowel single als mo3) staan op naam van Stanley Chapel, die al enkele jaren op rij zijn eigen records verbeterd. Graham Siggins heeft het MBLD record met een resultaat van 59/60, wat wil zeggen dat hij 60 Rubik's Cubes geblinddoekt heeft opgelost binnen het uur, waarvan 1 mislukt was. Volgende grafiek toont hoe records steeds moeilijker te verbreken zijn en nieuwe records slechts tienden van een seconde sneller zijn dan het vorige.



Geschiedenis van 3BLD single wereldrecords

Conclusie

Er zijn veel manieren om een kubus geblinddoekt op te lossen, maar slechts enkel methodes zijn geschikt om recordtijden mee te halen. Blindsolvers proberen hun techniek op elk vlak te optimaliseren om kostbare seconden te winnen. Zo heeft Graham Siggins bijvoorbeeld enkele jaren geleden de *Nod Don*-techniek bedacht: hierbij wordt de blinddoek aangedaan (don) met een knik van het hoofd (nod), waardoor beide handen op de kubus kunnen blijven en meteen overgegaan kan worden naar de uitvoering.

Wanneer we kijken naar de tijden van de records, is het duidelijk dat elke fractie van een seconde telt om kans te maken om het record te verbreken. In de uitvoeringsfase kan de meeste tijd gewonnen worden door de meest efficiënte algoritmes te gebruiken en deze goed in te oefenen.

Onderzoek doelgroep

Onderzoeksvraag

- Hoe groot is de doelgroep? Hoeveel speedcubers kunnen blindsolven?
- In welke leeftijdscategorie bevindt de doelgroep zich voornamelijk?
- Wat houdt speedcubers tegen om blindsolving te leren?
- Met welke blindsolve methodes is de doelgroep bekend?

Methodologie

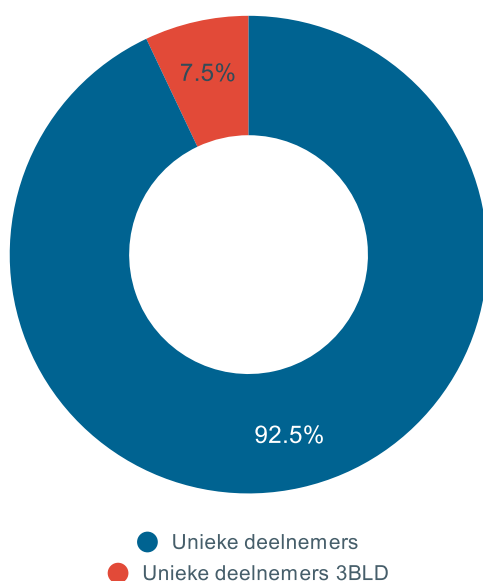
- Enquêtes
- Desk research

Voor dit onderzoek heb ik enquêtes geanalyseerd die gedeeld werden op de subreddit /r/cubers, één van de grootste Westerse online speedcubing communities. In 2015 vond de eerste zogenaamde 'Mega-Survey' plaats, een enquête waarin gevraagd werd naar onder andere demografie, kennis van speedcube methodes en puzzel collecties. Op dat moment telde de community 11.000 subscribers en hebben 466 speedcubers de enquête ingevuld. In mei 2021 werd de vijfde editie van de Mega-Survey (MS5) gedeeld. Gezien de community elk jaar sterk blijft groeien (meer dan 100.000 subscribers in 2021), was deze laatste enquête ook de grootste tot nu toe. In deze editie werden meer onderwerpen bevraagd en werd er meer in detail gegaan. Ook was dit voor de eerste keer in samenwerking met de Chinese community, die hun eigen online platformen hebben en normaal dus niet deelnemen aan deze enquêtes.

Ik heb ervoor gekozen deze bestaande Mega-Survey's te gebruiken, omdat deze relevante data bevatten voor het beantwoorden van de onderzoeksvragen. Ik heb de data waar nodig vergeleken met de voorgaande edities, waardoor ik ook mogelijke trends kan zien over de jaren heen.

Bijkomend werden relevante cijfers gehaald uit de export van alle WCA competitieresultaten die publiek gedeeld wordt op de officiële website.

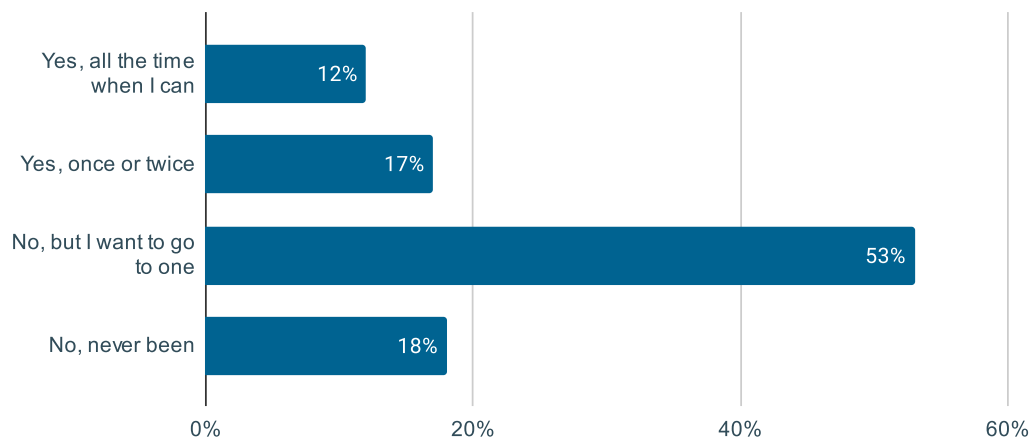
Resultaten



In deze sectie worden de vragen besproken van MS5 die relevante data bevatten voor het onderzoek. Waar relevant worden de resultaten vergeleken met de voorgaande edities van de Mega-Surveys.

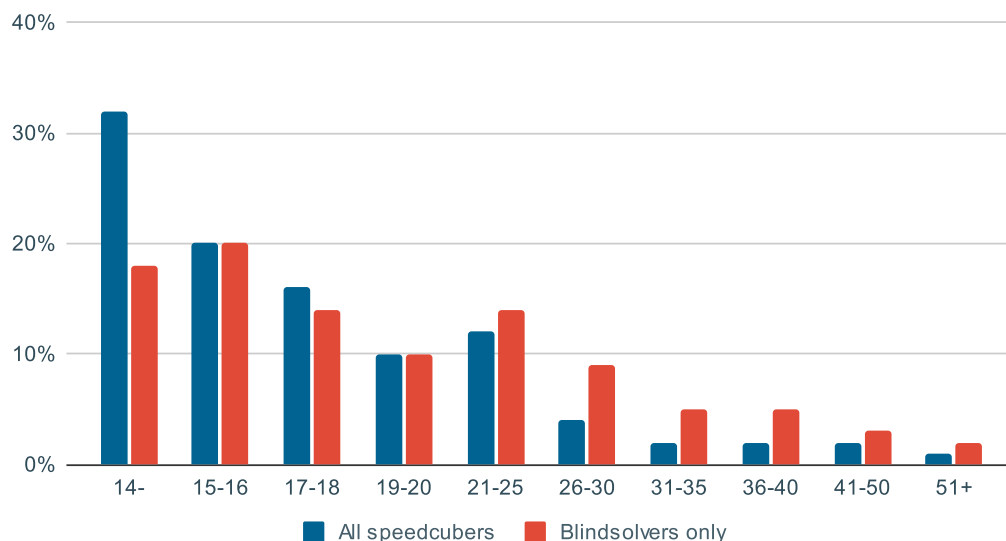
Demografie

Een duidelijk cijfer van hoeveel mensen speedcuben is er niet. Wel is geweten dat iets meer dan 150.000 speedcubers deelgenomen hebben aan een WCA competitie. We kunnen er vanuit gaan dat het werkelijke aantal speedcubers veel hoger ligt dan dit. Het percentage van deelnemers die participeren in het 3BLD event, bedraagt 7,5%.



Mega-Survey 5: Have you ever been to a WCA competition?

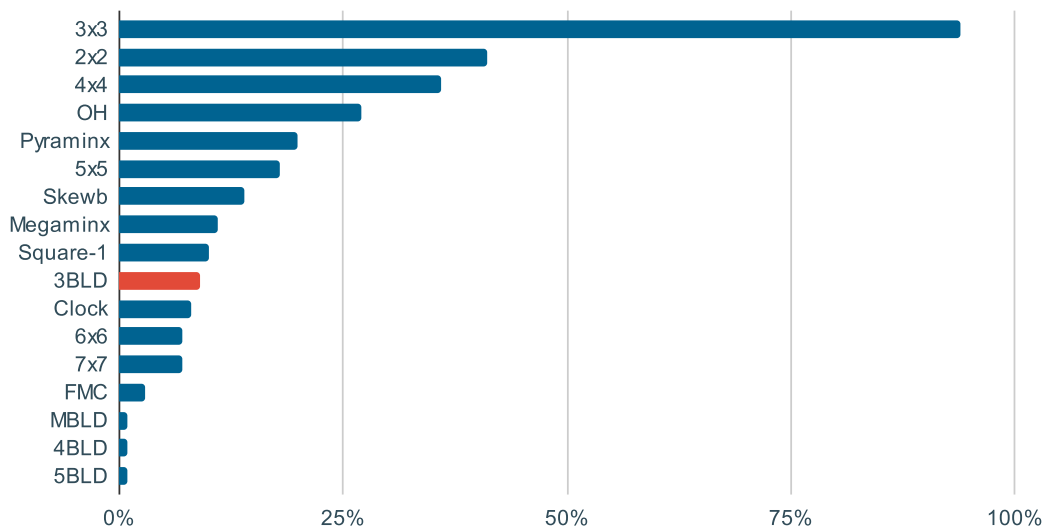
29% van de speedcubers die deelnamen aan MS5, antwoordde dat ze naar een competitie geweest zijn. Met deze data kan een grove schatting gemaakt worden dat er ongeveer 526.000 speedcubers zijn, waarvan 39.000 blindsolvers.



Mega-Survey 5: Age of the online speedcubing community

De gemiddelde leeftijd van speedcubers bedraagt 18 jaar. Meer dan de helft is jonger dan 16. De leeftijd van blindsolvers werd niet bevraagd, maar wel werd onderzocht welk percentage per leeftijdsgroep aan blindsolving doet. Door deze data te combineren met de gemiddelde leeftijd van alle speedcubers, werd de leeftijdsverdeling van blindsolvers gevonden (rode balken op de grafiek). Wanneer we dit vergelijken met de verdeling van alle speedcubers (blauwe balken op de grafiek), zien we dat de piek verschuift naar 15-16 jaar en een tweede piek bij 21-25 jaar groeit. Het aantal blindsolvers jonger dan 14 is opmerkelijk lager dan het totaal aantal speedcubers van dezelfde leeftijd. De interesse in blindsolving ligt het hoogst tussen de 31 en 35 jaar. Over het algemeen lijkt blindsolving aantrekkelijker te zijn op een relatief oudere leeftijd.

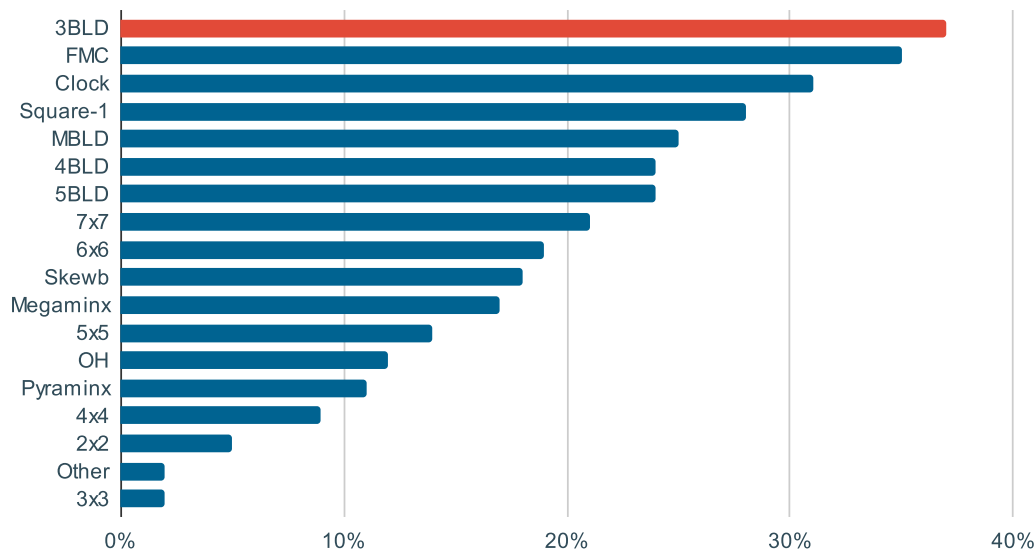
Interesse



Mega-Survey 5: Which events do you practice routinely?

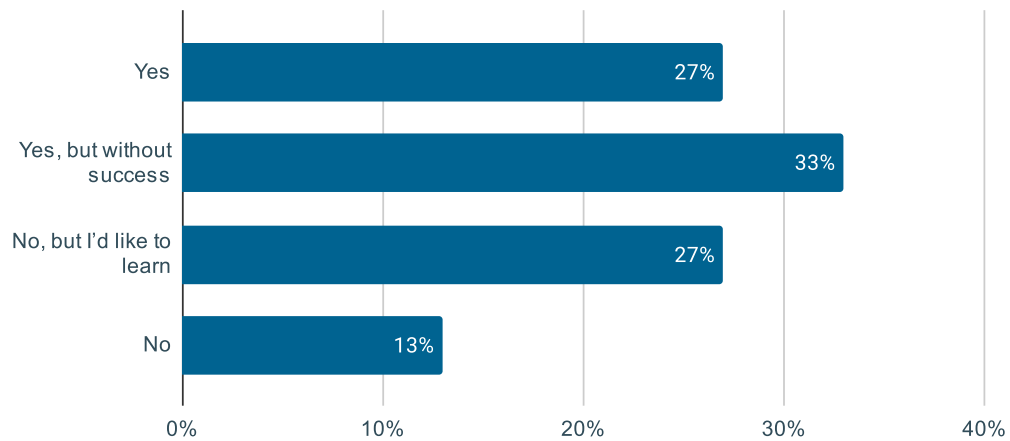
9% van de speedcubers oefent 3BLD routinematig. Dit leunt dicht aan bij de 7,5% die deelnemen aan 3BLD op competities. Dit cijfer lijkt ook stabiel te zijn in vergelijking met de resultaten uit de vorige Mega-Survey's. Vergeleken met de andere events, plaatst 3BLD zich ergens halverwege.

MBLD, 4BLD en 5BLD staan onderaan met 1%. Dit is een verwacht resultaat omdat het, ook voor geoefende blindsolvers, moeilijke events zijn die veel tijd en concentratie vragen. Het is dus minder eenvoudig deze events op regelmatige basis te oefenen.



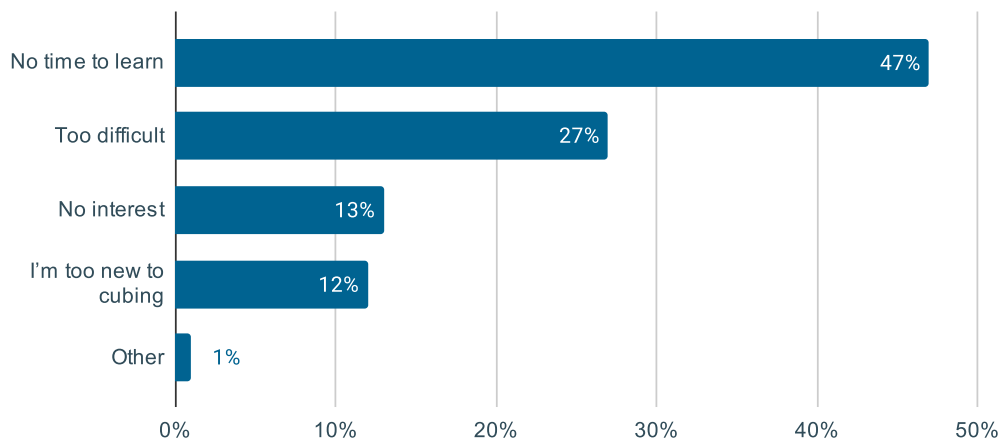
Mega-Survey 5: Which event would you like to try but have not yet?

Hoewel slechts 9% 3BLD oefent, toont 37% van de cubers interesse in het leren van het event. Hiermee is 3BLD het event waarvoor het meeste interesse is. Ook de andere BLD events scoren hoog: een kwart van de cubers hebben interesse in het leren van MBLD, 4BLD en 5BLD.



Mega-Survey 5: Do you know how to solve a cube blindfolded?

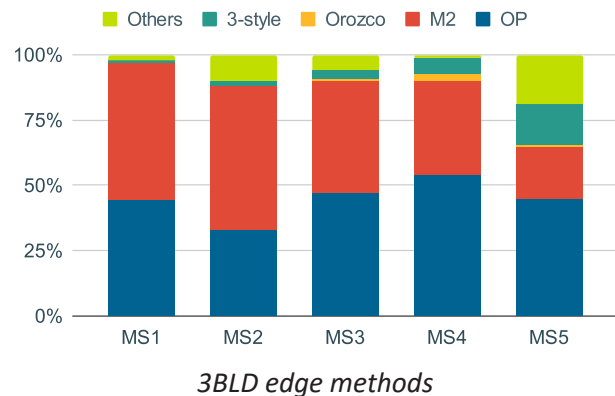
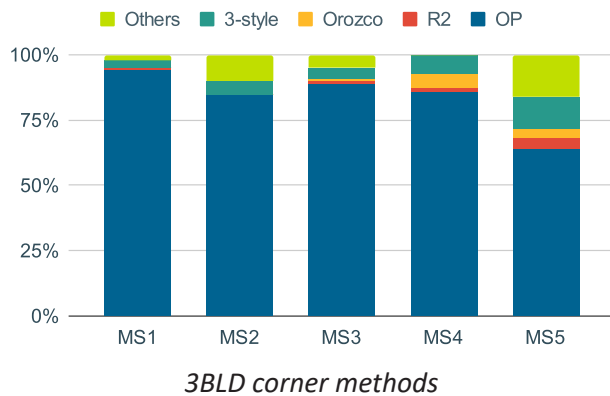
Uit MS5 blijkt ook dat 13% van de respondenten niet weet hoe ze een puzzel geblinddoekt oplossen en er ook geen interesse in heeft. Opvallend is de grote groep die weet hoe ze een puzzel geblinddoekt oplossen maar er nog niet in geslaagd is. Deze groep bevat zowel degenen die blindsolving aan het leren zijn, als degenen die weten hoe de methode in elkaar zit maar het nog niet zelf geprobeerd hebben.



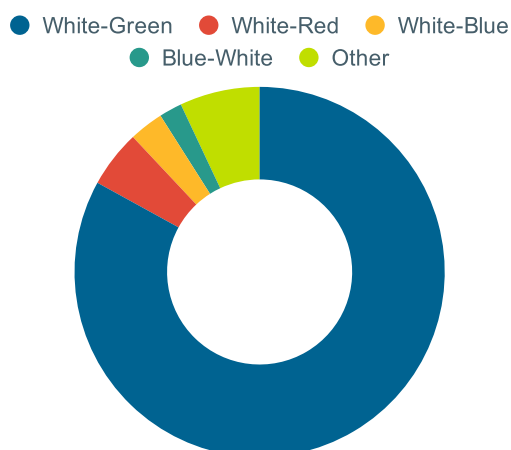
Mega-Survey 5: What is the main deterrent keeping you from learning BLD?

Tijdsgebrek blijkt de voornaamste reden te zijn om blindsolving niet te leren. De helft van de respondenten is van mening dat er veel tijd in kruipt. De tweede belangrijkste reden die wordt aangegeven is dat BLD leren te moeilijk lijkt – meer dan een kwart van de respondenten koos voor dit antwoord. Slechts 13% zegt geen interesse te hebben in het event, wat overeenkomt met het resultaat van de voorgaande vraag.

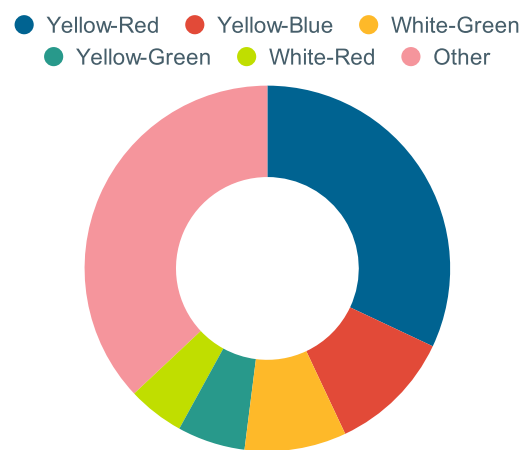
Techniek



Uit deze grafieken wordt duidelijk dat OP en M2 de populairste methodes zijn. Dit komt omdat OP de geprefereerde beginnersmethode is, terwijl M2 de volgende stap is om sneller te worden in het oplossen van de kubus. Vaak wordt OP als methode voor edges overgeslagen en leert men meteen M2. Ondanks overschaduwd te zijn door OP en M2, zien we dat het gebruik van de 3-style methode sterk gestegen is. In 2021 bedraagt het gebruik 12% voor corners en 6% voor edges, wat dubbel zoveel is vergeleken met het jaar daarvoor. Dit duidt op een stijgend aantal competitieve blindsolvers. Orozco, een methode die wel eens als opstap naar 3-style gebruikt wordt, blijft miniem aanwezig.



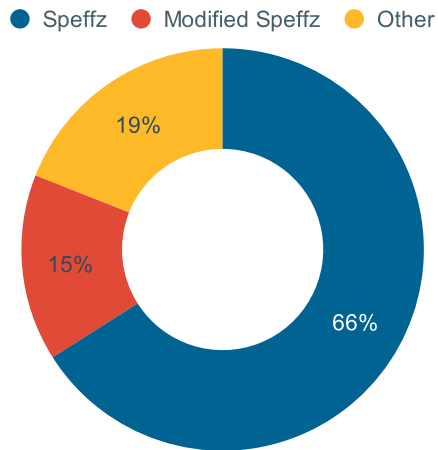
Mega-Survey 5: Most used color orientations (Western community)



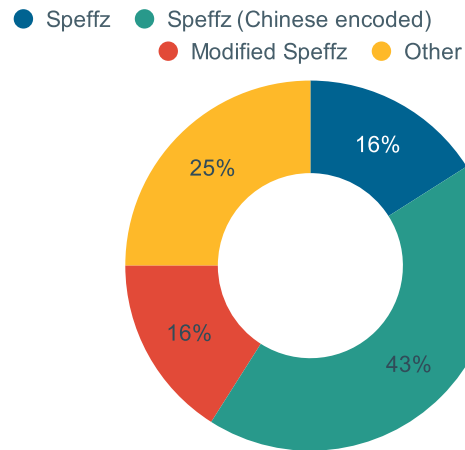
Mega-Survey 5: Most used color orientations (Chinese community)

De meest populaire oriëntatie binnen de Westerse community, wit (bovenaan) - groen (vooraan), wordt door 83% van de blindsolvers gebruikt. Deze oriëntatie is populair omdat het de standaard is volgens het WCA. Deze oriëntatie wordt bijvoorbeeld gebruikt om puzzels door elkaar gehaald. Opvallend is dat de populariteit sterk gestegen is de voorbije jaren. Slechts 39% koos voor deze oriëntatie in 2016. In 2017 en 2020 was het resultaat respectievelijk 47% en 66%.

De Chinese community hangt minder vast aan één oriëntatie. Geel-rood wordt het meest gebruikt, maar steekt minder uit boven de andere resultaten met 32%.



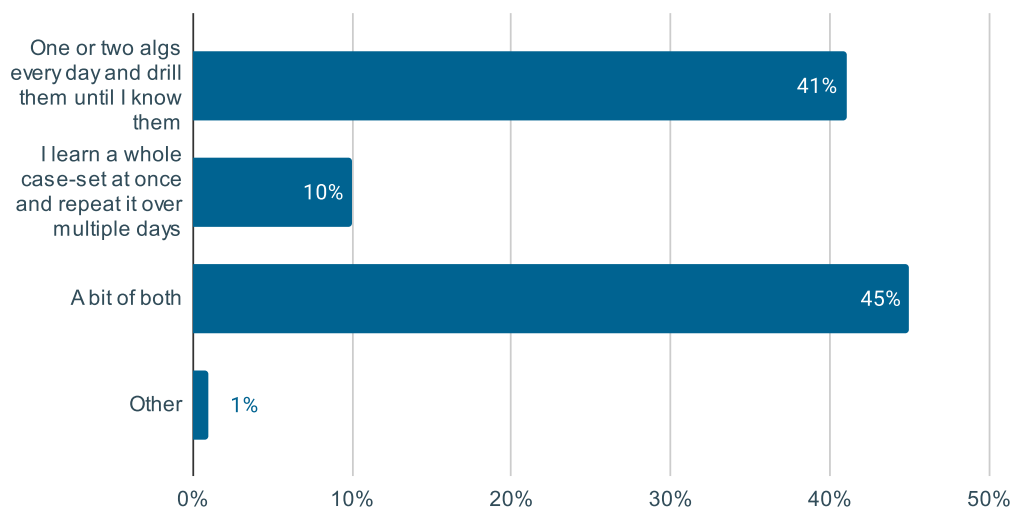
Mega-Survey 5: Most used lettering scheme (Western community)



Mega-Survey 5: Most used lettering scheme (Chinese community)

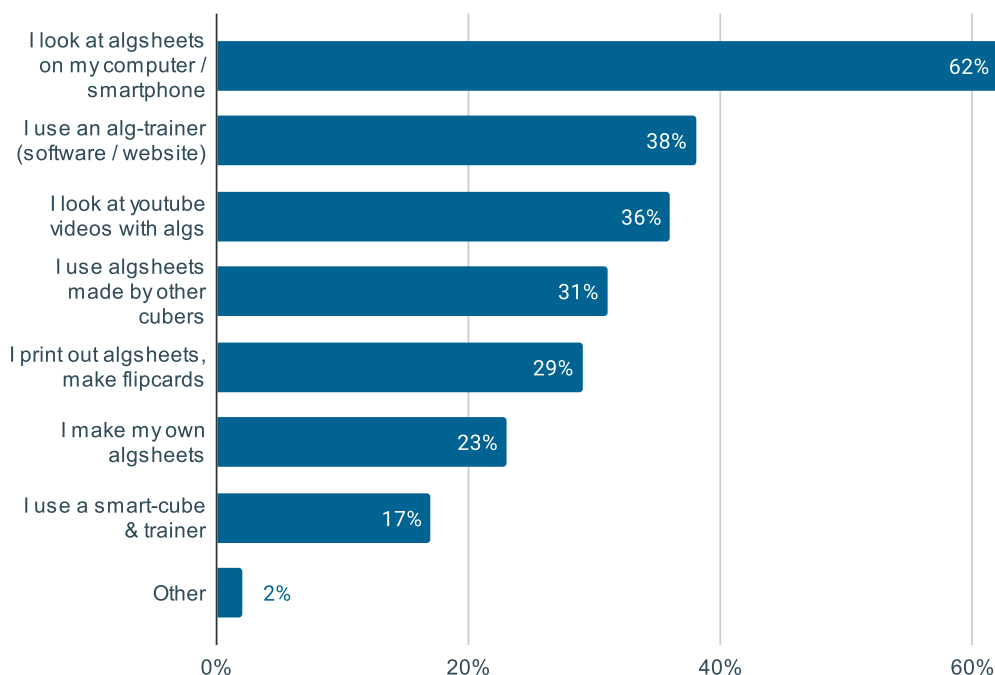
Speffz is duidelijk de populairste lettering scheme. Voor zowel de Westerse als Chinese community zijn de resultaten gelijkaardig. Het grote verschil is dat 43% van de Chinese blindsolvers een Chinese vertaling van Speffz gebruiken. Ook zien we dat bijna de helft die een andere lettering scheme gebruiken, slechts enkele aanpassingen doen aan Speffz. Vaak gaat dit om het aanpassen van moeilijke letters, zoals de letter Q, naar een eenvoudigere letter of klank.

Leermethodes



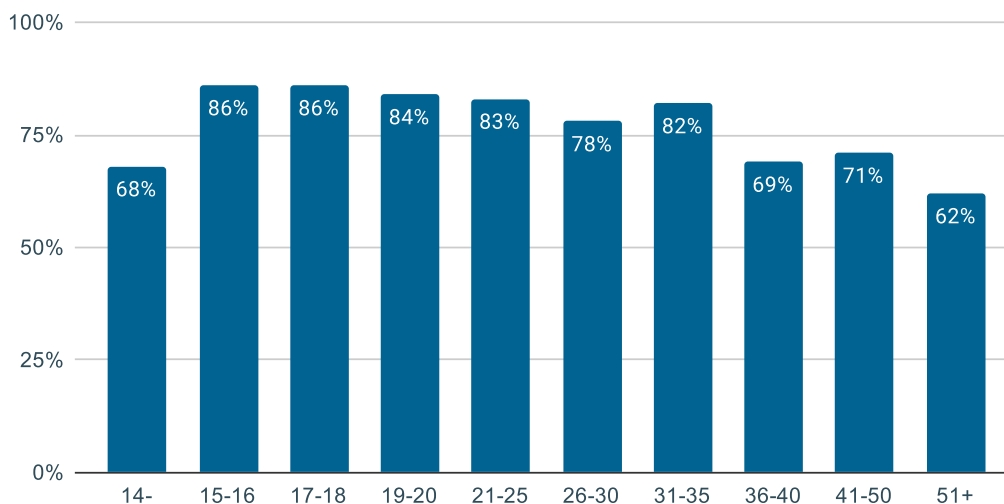
Mega-Survey 5: How do you learn new algorithms?

Enkele algoritmes per dag leren lijkt de populairste methode te zijn voor het leren van nieuwe sets. Het leren van een hele set in één keer wordt weinig gedaan, maar wel wordt dit vaak gecombineerd met de eerste methode.



Mega-Survey 5: What tools do you use to help yourself learn new algs?

Het gebruiken van algsheets is het populairste hulpmiddel voor het leren van algoritmes en komt in vier van de antwoorden voor. Ook trainers in de vorm van software- of webapplicaties worden gebruikt voor het inoefenen van deze algoritmes, en dit soms in combinatie met een smart cube. Verder is YouTube ook een veelgebruikt leermiddel.



Mega-Survey 5: Mobile usage by age

Als extra werd bevraagd naar het gebruik van smartphones. Het is niet onbelangrijk om te weten dat het mobile gebruik hoog ligt, wat een invloed zal hebben op de vorm van het werkstuk.

Conclusie

Vergeleken met de leeftijdsverdeling van alle speedcubers, is blindsolving populairder bij de iets oudere leeftijden. Vooral bij de grote groep cubers onder de 14 jaar is het minder populair.

Ondanks de hoge interesse in blindsolving, oefenen iets minder dan 10% routinematig. Dit is verrassend lager dan het aantal personen die een 3x3 kubus geblinddoekt kunnen oplossen, of

interesse hebben in het leren ervan. Er lijkt een drempel te zijn om meer te verdiepen in blindsolving. Dit kan mede te wijten zijn aan de moeilijkheid van het event, de angst om te falen, en het gebrek aan goede leermiddelen in vergelijking met andere events. Dit laatste wordt ook bevestigd door het feit dat tijdsgebrek de belangrijkste factor is die cubers ervan weerhoudt om blindsolving te leren.

Verschillende standaarden, zoals de wit-groen oriëntatie en het Speffz lettering scheme, worden door meer en meer blindsolvers gebruikt. Ondanks de voordelen hiervan, kan er niet vanuit gegaan worden dat iedereen deze standaarden gebruikt en is flexibiliteit noodzakelijk voor de inclusiviteit.

Onderzoek concurrentie

Onderzoeksvraag

- Welke tools bestaan er om algoritmes bij te houden?
- Welke tools bestaan er om algoritmes te leren?
- Welke features zijn belangrijk hiervoor?
- Welke tools bestaan er specifiek voor blindsolvers?

Methodologie

Desk research

De beginnersmethode voor het leren van blindsolving is vrij eenvoudig aan te leren en hiervoor bestaan verschillende goede tutorials. Om deze reden bestaan er weinig tools voor het leren van deze methode, en zal ik mij in dit onderzoek focussen op tools die gericht zijn op geavanceerde methodes zoals 3-style.

Resultaten

Spreadsheets

Uit het onderzoek naar het onderwerp is gebleken dat 3-style bestaat uit 818 algoritmes. Deze vormen de basis en vaak worden uitbreidingen geleerd om bepaalde gevallen efficiënter op te lossen. Om deze algoritmes bij te houden, wordt vaak gebruik gemaakt van spreadsheets. Doordat 3-style algoritmes een combinatie van twee cubies oplossen, is een tabel de ideale manier van weergeven. Via Google Sheets worden deze spreadsheets publiek gedeeld, zodat nieuwe blindsolvers kunnen zien welke algoritmes gebruikt worden door de meer ervaren blindsolvers.

Graham's Full 3Style Comm Series + more																											File Edit View Search Format Data Tools Extensions Help	Y - 100% View history	Share
K27		A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z		
1	first/last	A (UBL)	B (UBR)	C (UBL)	D (UBL)	E (FUL)	F (FUL)	G (FUL)	H (FUL)	I (FUL)	J (DFR)	K (DBR)	L (DBL)	M (RUB)															
2	A (UBL)	[R'B'R : [R'D,R,U] : [R'B'R : [R'D,R,U																											

Graham Siggins' spreadsheet

Bldbase.net

Algdb.net en Speedcubedb.com








[AlgoDx.net](#)
[Home](#)
[Users](#)

Home

3x3

PLL

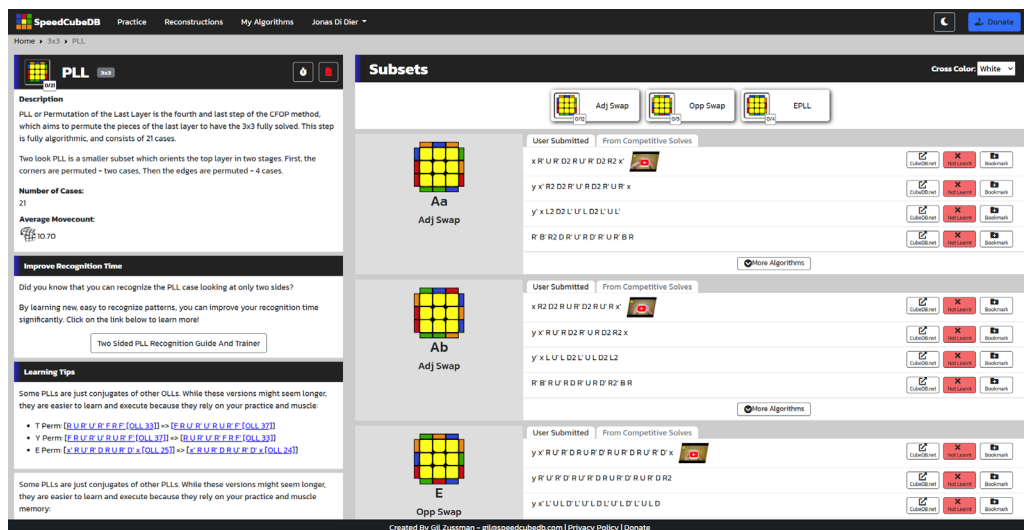
Filter by case name

Case	Algs
Aa	 <pre> F U R' D2 U F R' D2 R2 x R' U R' D2 R U F R' D2 R2 R' F R' R2 R' F R' R2 R2 y x R2 D2 R U F R D2 R U R x </pre>
Ab	 <pre> x R2 D2 R U R' D2 R U R x' F R' D2 R U R' D2 R U F R x' y x R U R D2 R U R D2 R2 x y' x U F L D2 L' U L D2 L2 </pre>
E	 <pre> y x' R U R' D R U R' D' R U R' D R U R' D' x R2 U R' y R U R' U R U R' U R U R' y R U R'2 z D2 R2 F R U R' U R U R' U R U R' F R2 U2 z' R2 U R2 U D R2 U R2 U R2 U R'2 U R2 U2 R2 </pre>
F	 <pre> y R' U F R U R' U R' F R2 U' R' U R U R' R z R' U2 R2 R' d' F R2 U' R' U R' U R' F R' U R U R2 U' U' U R U R' F R2 M' U2 L F R U2 r' U' F R2 U2 R2 </pre>
Ga	 <pre> R2 u R' U R' R'2 y' R U R R2 U R' U R' R2 D U R' U R D R2 u R' U R' R'2 U' U' D' R2 U R' U R' R2 U' D R U R </pre>
Gb	 <pre> R' U' R y R2 u R' U R U R' R2 R' U R U D' R2 U R U R U R' R2 D y' F U F R2 u R' U R U' R2 R' d' F R2 u R' U R U' R2 </pre>
Gc	 <pre> R2 u' R' U R R' u R2 y R U R' </pre>

19

Deze site houdt echter geen algoritmes bij voor blindsolving. Aangezien de development gestopt is in 2018, lijkt het ook onwaarschijnlijk dat dit nog toegevoegd zal worden.

'Speed Cube Database', ofwel SpeedCubeDb, is een nieuwe database die probeert de gebreken van Algdb op te lossen. De site werd ontwikkeld in 2020 naar aanleiding van een video van Jayden McNeill, een populaire speedcuber, waarin hij zijn idee vertelt van een betere database waarbij algoritmes automatisch geverifieerd worden en dus geen moderators nodig zijn. Naast de automatische verificatie, werden ook allerlei andere features toegevoegd. Speedcubedb ondersteunt meer puzzels, heeft het een ingebouwde trainer om algoritmes te timen en toont reconstructies van solves van bekende speedcubers.



Speedcubedb.com

Ondanks de vele features, is hier ook niks terug te vinden voor blindsolvers. Verder heeft de website een drukke en onduidelijke UI en is er geen optimalisatie voor tragere netwerken. Hoewel Speedcubedb meer functionaliteit biedt dan Algdb, maakt dit de site minder aangenaam om te gebruiken.

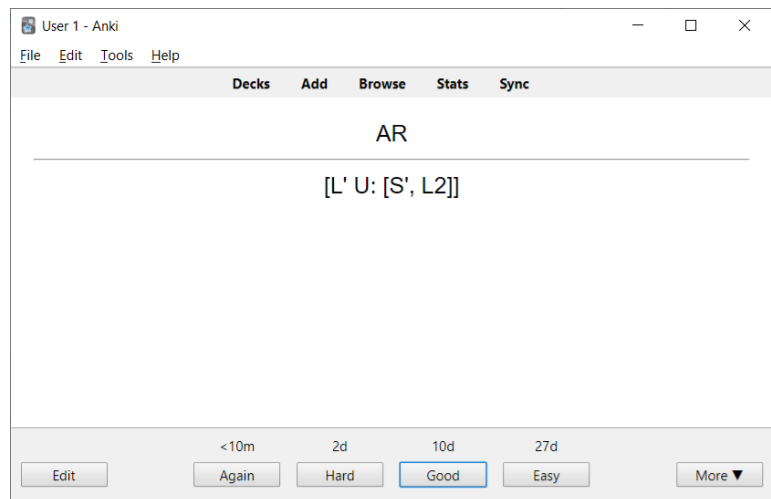
BLD Support Group

BLD Support Group is een Facebook groep waarin blindsolvers bij elkaar terecht kunnen voor allerlei vragen en discussies. De groep is aangemaakt in 2013 en bevat momenteel iets meer dan 3400 leden. Een groot deel van de actieve leden zijn blindsolvers van hoog niveau, die met hun inzichten bijdragen tot betekenisvolle discussies en hulp kunnen bieden op vragen. Vaak is er vraag naar goede alternatieven voor een bepaald algoritme. Frequent gaan deze vragen gepaard met een poll waarop de leden op het beste algoritme stemmen. Het valt op dat, door een gebrek aan een goede database zoals Algdb voor blindsolving, deze vragen vaak gesteld moeten worden.

Flashcards

Naast het bijhouden van algoritmes, is het uiteraard ook nodig om deze te leren. De manier van leren hangt af van persoon tot persoon, maar vaak worden hiervoor extra hulpmiddelen gebruikt. Zoals gebleken uit het doelgroeponderzoek maakt 38% gebruik van trainers, en 29% print flashcards. Aangezien de algoritmesets veel groter zijn bij blindsolving, zullen deze cijfers voor blindsolvers nog hoger liggen.

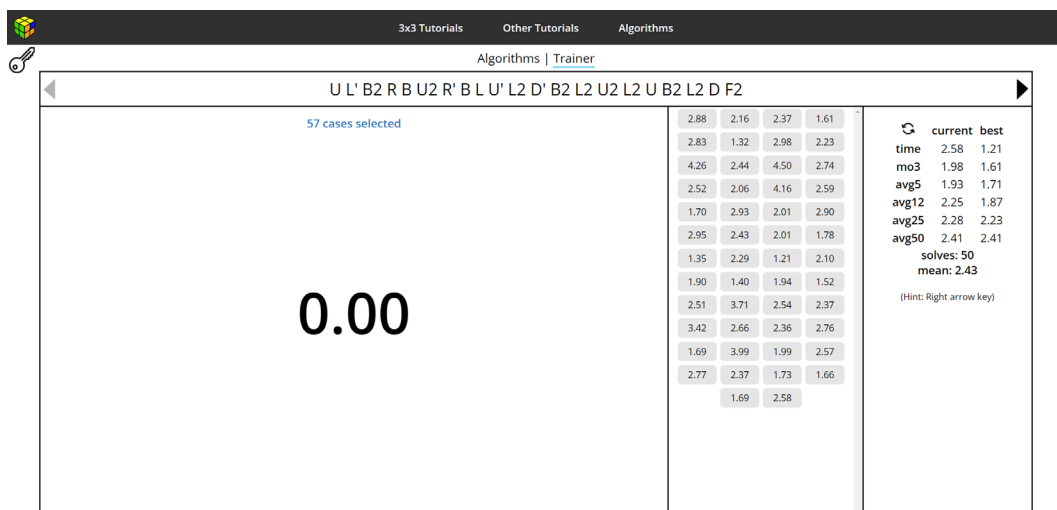
Anki is een open source flashcard tool die door blindsolvers gebruikt wordt voor het leren van 3-style algoritmes. Het maakt gebruik van de spaced repetition leertechniek, wat betekent dat nieuwe en moeilijke flashcards vaker getoond worden dan oude en gemakkelijkere flashcards. Anki is een krachtige tool met veel functionaliteit, maar aangezien het niet afgestemd is op blindsolving, voelen de meeste functies overbodig aan, wat de UX belemmert.



Anki

Een groot probleem met Anki en andere flashcard trainers is dat er geen link is tussen de data in de spreadsheets en de data in de trainer. Dit zorgt ervoor dat wijzigingen aan algoritmes op twee plaatsen moeten gebeuren, wat in de realiteit betekent dat de datasets in Anki verouderd en niet meer van toepassing zijn. Dit wordt bevestigd in een video van Jack Cai (voormalig wereldrecordhouder) over Anki, waarin te zien is hoe enkele flashcard decks gelabeld zijn als 'old'. Daarnaast is het één voor één manueel toevoegen van de data een enorme karwei. Hoewel spreadsheets het mogelijk maken om de data weer te geven op een manier zodat het eenvoudiger geïmporteerd kan worden in de flashcards trainer, blijft het een omslachtige manier van werken.

Jperm.net is een website van Dylan Wang (gekend als 'J Perm' op YouTube) waarop hij allerlei tutorials en algoritmesets deelt. Wat interessant is aan deze website, is dat elke set van algoritmes gelinkt is aan een trainer. Verschillend met tools als Anki, is dat deze trainer de nadruk legt op het timen van de uitvoering van algoritmes. Bovenaan wordt een scramble getoond die de user toepast op zijn kubus, waarna hij het juiste algoritme zo snel mogelijk uitvoert om de kubus op te lossen.



Jperm.net trainer

In het overzicht van algoritmes staat naast elke case de beste en gemiddelde tijd van de uitvoering. Dit geeft een duidelijk overzicht voor de user welke algoritmes meer oefening nodig hebben of aangepast moeten worden. Voor dit laatste is het namelijk ook mogelijk om een algoritme te vervangen door erop te klikken en een alternatief te kiezen uit de lijst. De algoritmes kunnen ook

gemarkeerd worden met een kleur (geel of groen) om bij te houden welke al geleerd zijn en welke niet. Dit is echter puur visueel – in de trainer is hier geen optie voor voorzien (er kan tijdens het trainen niet aangeduid worden dat de user een algoritme vergeten is). Hoewel de website tutorials bevat over blindsolving, is de trainer niet beschikbaar voor blindsolving algoritmes.

Ook Speedcubedb bevat een soortgelijke trainer, maar is beperkter dan de trainer op Jperm.net en voegt weinig nieuws toe.

Conclusie

Bij gebrek aan alternatieven zijn spreadsheets de meest gebruikte manier voor blindsolvers om hun algoritmes bij te houden. Bestaande databases zoals Algdb en Speedcubedb zijn nog niet ontwikkeld voor blindsolving, ondanks de vele voordelen die ze bieden.

Leerhulpmiddelen zijn gelimiteerd tot flashcards, die niet afgestemd zijn op blindsolving. De disconnectie tussen de spreadsheets en trainers zorgen ook voor extra werk, wat er vaak tot leidt dat de data in de trainers verouderd raken.

Het gebrek aan goede tools voor blindsolving lijkt te wijten aan het feit dat dit een kleine niche is, ondanks dat de community hier enorme baat bij zou hebben.

Minder belangrijk maar ook opvallend is dat vele speedcubing websites en tools een verouderde UI en onvriendelijke UX hebben. Een moderne, responsive layout zou ervoor kunnen zorgen dat users meer gemotiveerd zijn om de website te bezoeken en te gebruiken.

Onderzoeksconclusies

Uit het onderzoek is duidelijk geworden dat blindsolving events complexer zijn dan de andere WCA events. Ondanks dit feit zijn er weinig hulpmiddelen voor het leren of optimaliseren van blindsolving methodes. Speedcubers lijken er over het algemeen wel interesse in te hebben, maar worden tegengehouden door tijdsgebrek of hebben het idee dat het te moeilijk is om te leren. Blindsolvers die hun methodes willen verbeteren, maken gebruik van ineffectieve tools.

Voor het werkstuk zal het belangrijk zijn om mij te richten op de specifieke noden van blindsolvers die andere, meer algemene tools niet bieden. Uit het doelgroeponderzoek is gebleken dat tijdsgebrek de voornaamste reden was om blindsolving niet te leren, ondanks de interesse. Ook werd er vaak gedacht dat blindsolving te moeilijk is om te leren. In het werkstuk zal dus een manier gevonden moeten worden om de vele algoritmes in kleine stukken aan te leren, om zowel tijd te besparen als de drempel te verlagen. Ook is duidelijk geworden dat niet iedere blindsolver op exact dezelfde manier te werk gaat. De verschillen tussen onder andere algoritmenotaties, lettering schemes en kleuroriëntatie zorgen ervoor dat de uniforme weergave voldoende aanpasbaar moet zijn voor elke user.

PRODUCTIE

Omschrijving

Voor mijn bachelorproef heb ik BlindsolveDB (blindsolvedb.com) ontwikkeld, een webapplicatie die een oplossing biedt voor de problemen die uit het onderzoek duidelijk zijn geworden. De applicatie bestaat voornamelijk uit twee grote onderdelen: de sheets en de trainer.

De sheets zijn een collectie van algoritmesets gegroepeerd per categorie, die elk een overzicht hebben van alle cases van de set. Voor elke case wordt het meest populaire algoritme weergegeven. De detailpagina van een case toont alle mogelijke algoritmes om die case op te lossen waarvan de user het algoritme selecteren dat hij of zij gebruikt. Naast de weergave van de meest populaire algoritmes, is er voor een ingelogde user ook de mogelijkheid om hun eigen geselecteerde algoritmes te zien. Ook is er een optie om per case een nieuw algoritme toe te voegen, en kunnen algoritmes gerapporteerd worden indien deze niet lijken te kloppen.

De trainer is een feature die helpt bij het leren van nieuwe sets en het herhalen van gekende sets. Per set kan een *learn* of een *review* sessie gestart worden, waarbij maximaal 10 algoritmes per keer aangeleerd of herhaald worden. Tijdens een sessie worden verschillende cases getoond, waarna de user telkens het bijhorende algoritme uitvoert op de kubus die hij in de hand heeft. Na het uitvoeren wordt de duurtijd weergegeven van de uitvoering, en kan de user het resultaat aanduiden (*good*, *bad* of *failed*). Op basis van de resultaten wordt op de server bijgehouden wanneer een algoritme opnieuw gereviewd moet worden, wat weergegeven wordt op de trainer pagina.

Naast de sheets en trainer is er een profielpagina waar de user de color scheme en lettering scheme kan aanpassen. Ook is er een admin pagina waar alle reports bekeken en opgelost kunnen worden.

Aanvullend op de webapplicatie, werd de library magic-cubes ontwikkeld om algoritmes te kunnen valideren en visualiseren. Deze library bevat de *Algorithm* klasse, wat een eigen implementatie is van een parser voor 3x3 algoritmes. De library heeft ook een *Cube* klasse die de state van een Rubik's kubus simuleert. Verder is er ook de *CubeModel* klasse die wordt toegepast op de webapplicatie voor het visualiseren van een kubus.

Scope

BlindsolveDB is bedoeld voor speedcubers die enige ervaring hebben met blindsolving, omdat geavanceerde blindsolving methodes meer nood hebben aan een goede tool. De doelgroep heeft ervaring met methodes als OP en M2 en is ofwel geïnteresseerd in het leren van 3-style, ofwel reeds vertrouwd met 3-style. In deze laatste groep wordt de applicatie dan voornamelijk gebruikt om algoritmes bij te houden en optimaliseren.

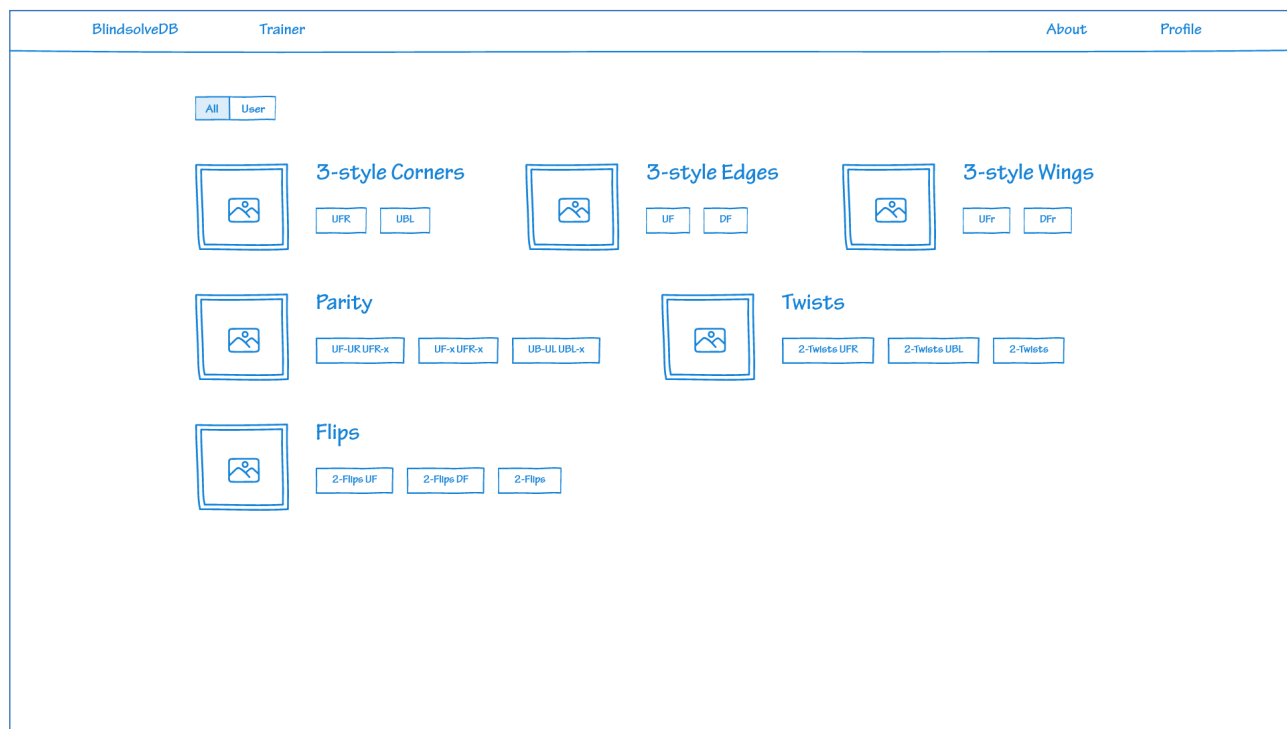
De focus ligt enkel op de uitvoeringsfase van een blindsolve. Memorisatie is een heel aparte uitdaging met andere noden en behoeften, wat de scope van dit werkstuk te ruim zou maken.

Planning

- Wireframes
- Data model
- Frontend design
- Features: Sheets – Trainer – Profiel – Admin dashboard
- Deployment

Voorbereidingen

Wireframes



BlindsolveDB

Trainer

About

Profile

3-Style Corners — UFR

< Return

Popular

User

	A	B	D	E	F	G	H	I	K	L	N	O
A		[U, R D R]	[F: [u2, F R F R]]	[R D R, u2]	[U: [R D R', u2]]	[D' D U: [R D R', u2]]	[D U: [R D R', u2]]	[D: [R D R', u2]]	[R: [U', R D R]]	[U: [R D R', u2]]	[R D R, u2]	[R U D: [R D R', u2]]
B	[R F R: [U', R D R]]		[F: [U, R D R]]	[R2: [D', R U R]]	[R D R, U]	[U, R D R]	[R2 U: [D', R U R]]	[U: [R D R, U]]	[D: [R D R, U]]	[U', R D R]	[R D R, U]	[R2 U: [R U R', D]]
D	[R D R U: [R D R', U]]	[R F R: [U, R D R]]		[R D R, U]	[U: [R D R', U]]	[D: [R D R', U]]	[U: [R D R', U]]	[U: [R D R', U]]	[R: [U2, R D R]]	[U: [R D R', U]]	[R D R, U]	[R: [R D R', U]]
E	[R2: [D, R U R]]	[U R: [u2, R D R]]	[U R F: [D, R U R]]		[R D: [R D R', u2]]	[U: [R D R', u2]]	[U D R: [U', R D R]]	[R U: [R D R', u2]]	[R: [U', R D R]]	[U R U2: [R D R', D]]	[U R U2: [R D R', u2]]	[R: [F L F]]
F	[F: [R D R', u2]]	[R: [U', R D R]]	[R U: U]	[R U: [D, R U R]]		[R: [F, R U R U]]	[U R U: [D, R U R]]	[U R D R: [D, R U R]]	[R U: [R D R', u2]]	[R F: [R U R, D]]	[U F, F]	
G	[u2, R D R]	[U, R D R]	[U', R D R]	[U R: [R D R, U]]	[R U: [R U R', D]]		[D R: [F2, R D R D]]	[U: [D, R U R]]	[U R U: [R D R', u2]]	[U D: [R U R', D2]]	[U R D: [R U R', D2]]	[R U: [R U R', D]]
H	[U: [R D R', u2]]	[R D R, U]	[u2: [R D R', u]]	[R D: [U, R D R]]	[U R U: [R U R', D]]	[R D: [F, D R D R]]		[U: [D, R U R]]	[D2: [R U R', D]]	[U R: [R D R', u2]]	[R D: [R U R', D2]]	[D2: R U R]
I	[R D R U R: [F, R U R U]]	[R2 U: [R U R', D]]	[U: [R D R', u2]]	[R D: [F2, D R D R]]	[R U D: [R D R', u2]]	[R: [U', R D R]]	[R U: [R D R', u2]]		[R D U: [R D R', u2]]	[D R U: [R D R', u2]]	[R U D: [R D R', u2]]	[R: [U', R D R]]
K	[U D: [R D R', u2]]	[D: [U, R D R]]	[D: [U', R D R]]	[D U R: [R D R', U]]	[U R U: [F, U R U R]]	[U: [R U R', D]]	[U U: [R U R', D]]	[R: [F2, R D R D]]		[U R U: [R D R', u2]]	[U: [R U R', D]]	[D R U: [R U R', D]]
L	[U: [R D R', u2]]	[D: [U, R D R]]	[U: [U', R D R]]	[u2: [R D R', u]]	[D R U: [R U R', D]]	[D: [R U R', D]]	[U: [R D R', u]]	[D R D: [F2, D R D R]]	[U R U: [R D R', u2]]	[U R U: [R D R', u2]]		[R F: [R U R', D]]
N	[R U D R: [D, R U R]]	[R: [U', R D R]]	[R: [U', R D R]]	[U R U: [R U R', D]]	[U R U: [R D R', u2]]	[U R: [u2, R D R]]	[R: [U', R D R]]	[R D U: [R D R', u2]]	[D R U: [R D R', u2]]	[R U D: [R D R', u2]]	[U R U: [R D R', u2]]	[U R2 U: [D', R U R]]
O	[U: [R D R', u2]]	[R D R', u]	[u2: [R D R', u]]	[U R F R: [R D R', U]]	[D R: [U, R U R', F]]	[U D: [R U R', D2]]	[R D: [R U R', D2]]	[R D U: [R D R', u2]]	[D: [U', R U R]]	[R D: [R U R', D]]	[U R U: [R D R', u2]]	
P	[u2, R D R]	[U, R D R]	[U', R D R]	[U R U: [R D R', U]]	[R U: [R U R', D]]	[U R D: [R U R', D2]]	[R U R', D2]	[D R U: [R D R', u2]]	[R U R', D]	[R: [U', R D R]]	[R D: [U, R D R]]	[R U: [R U R', D]]
Q	[R U D: [R D R', u2]]	[R D: [R D R', u]]	[F L F, F]	[R F: [D, R U R]]	[R U: [D, R U R]]	[R, U L U]	[R D U: [R D R', u2]]	[D R U: [D, R U R]]	[R F: [D, R U R]]	[U R U: [D, R U R]]	[R U: [D, R U R]]	[R U R U: [R D R', u]]
R	[R2 U: [D, R U R]]	[U R2: [D, R U R]]	[F, U F]	[R D R: [U, R D R]]	[D R R F: [R U R', D]]	[R: [U', R D R]]	[U R U: [R D R', u2]]	[D R: [R D R', D]]	[U R2 U: [R U R', D]]	[R D: [R U R', D]]	[R2: [U, R D R]]	[U U: [R U R', D]]
S	[U D: [R D R', u2]]	[D: [R D R', u]]	[D: [U', R D R]]	[D U R: [R D R', u]]	[F, u2 F]	[u2: [R U R', D]]	[D R D: [F2, D R D R]]	[U: [D2, R U R]]	[D R D: [R U R', D2]]	[U R U: [R D R', u2]]	[U: [D, R U R]]	[U R D: [R D R', u]]
T	[U D: [R D R', u2]]	[D: [R D R', u]]	[D: [U', R D R]]	[u2: [R D R', u]]	[D R U: [R U R', D]]	[R D R2: [U, R D R]]	[D: [R U R', D]]	[R U: [R D R', u2]]	[R U R U: [R U R', D]]	[R U: [R D R', u2]]	[R: [U', R D R]]	[R U R U: [R D R', u]]
U	[R F R U: [R D R', u2]]	[R F: [D, R U R]]	[U: [F2, U R U R]]	[D R: [U, R D R]]	[U R U: [D, R U R]]	[D R U R: [u2, R D R]]	[R D U: [R D R', u2]]	[R: [F2, R U R U]]	[U R: [u2, R D R]]	[U R: [U, R D R]]	[R D: [R D R', u]]	[D R U: [D, R U R]]
V	[u2, R D R U R D R]	[U, R D R U R D R]	[U', R D R U R D R]	[U R D: [R D R', u]]	[U D R U: [D, R U R]]	[U R D: [R D R', u]]	[U R D R: [u2, R D R]]	[R: [U', R D R]]	[R U R: [u2, R D R]]	[U R R: [u2, R D R]]	[D R: [F2, R U R U]]	[U R U: [D, R U R]]
W	[R D R: [u2, R D R]]	[R D R U R D R, U]	[R D R: [U', R D R]]	[R R: [U, R D R]]	[U R R2: [U, R D R]]	[R D R: [U, R D R]]	[D R: [u2, R D R]]	[D R D: [R D R', u]]	[R R2: [R U R', D]]	[U R U D: [R D R', u2]]	[D R U R: [u2, R D R]]	[D R U R: [u2, R D R]]
X	[R U D R: [D, R U R]]	[R B: [R U R', D]]	[R D R: [D, R U R]]	[R: [U, R D R]]	[U R U R: [D, R U R]]	[D R: [F2, R U R U]]	[R2 U: [R U R', D]]	[R D R: [D, R U R]]	[D R D: [R D R', u]]	[D U R: [u2, R D R]]	[U D R D: [R D R', u]]	[R2: [R U R', D]]

BlindsolveDB

Trainer

About

Profile

AB

< 3-Style Corners — UFR

Algorithm	Users	Use
[R' B' R; [U', R D R']]	43	<input type="radio"/>
[R' B' R2; [D, R' U' R']]		<input type="radio"/>
[U; [U, R' D2 R']]	8	<input checked="" type="radio"/>
[x R; [U, R' D2 R']]		<input type="radio"/>
[x R2; [R U R', D2]]		<input type="radio"/>
[R' D' R U; [U, R' D' R']]	1	<input type="radio"/>

+ Add algorithm

BlindsolveDB

Trainer

About

Profile

Review

3-style Corners — UFR

21%80/378

Review (26)

Parity — UF-UR UFR-x

100%21/21

Review (7)

3-style Edges — UF

90%396/440

Learn

Twists — 2-Twists UFR

100%14/14

Train

Flips — 2-Flips UF

100%11/11

Train

BlindsolveDB

Trainer

About

Profile

3-style Corners — UFR

21%80/376

Learn

Review (26)

A

B

D

E

F

G

H

I

K

L

N

O

P

Q

R

S

T

U

Case	Algorithm	Average
AB	[R' B' R; [U, R D R]]	1.83
AD	[R' B' R; [U, R D R]]	1.83
AF	[R' B' R; [U, R D R]]	1.83
AG	[R' B' R; [U, R D R]]	1.83
AH	[R' B' R; [U, R D R]]	1.83
AI	[R' B' R; [U, R D R]]	1.83
AK	[R' B' R; [U, R D R]]	1.83
AL	[R' B' R; [U, R D R]]	1.83
AN	[R' B' R; [U, R D R]]	1.83
AO	[R' B' R; [U, R D R]]	1.83
AP	[R' B' R; [U, R D R]]	1.83
AQ	[R' B' R; [U, R D R]]	1.83
AS	[R' B' R; [U, R D R]]	1.83
AT	[R' B' R; [U, R D R]]	1.83
AU	[R' B' R; [U, R D R]]	1.83
AV	[R' B' R; [U, R D R]]	1.83
AW	[R' B' R; [U, R D R]]	1.83
AX	[R' B' R; [U, R D R]]	1.83

BlindsolveDB

Trainer

About

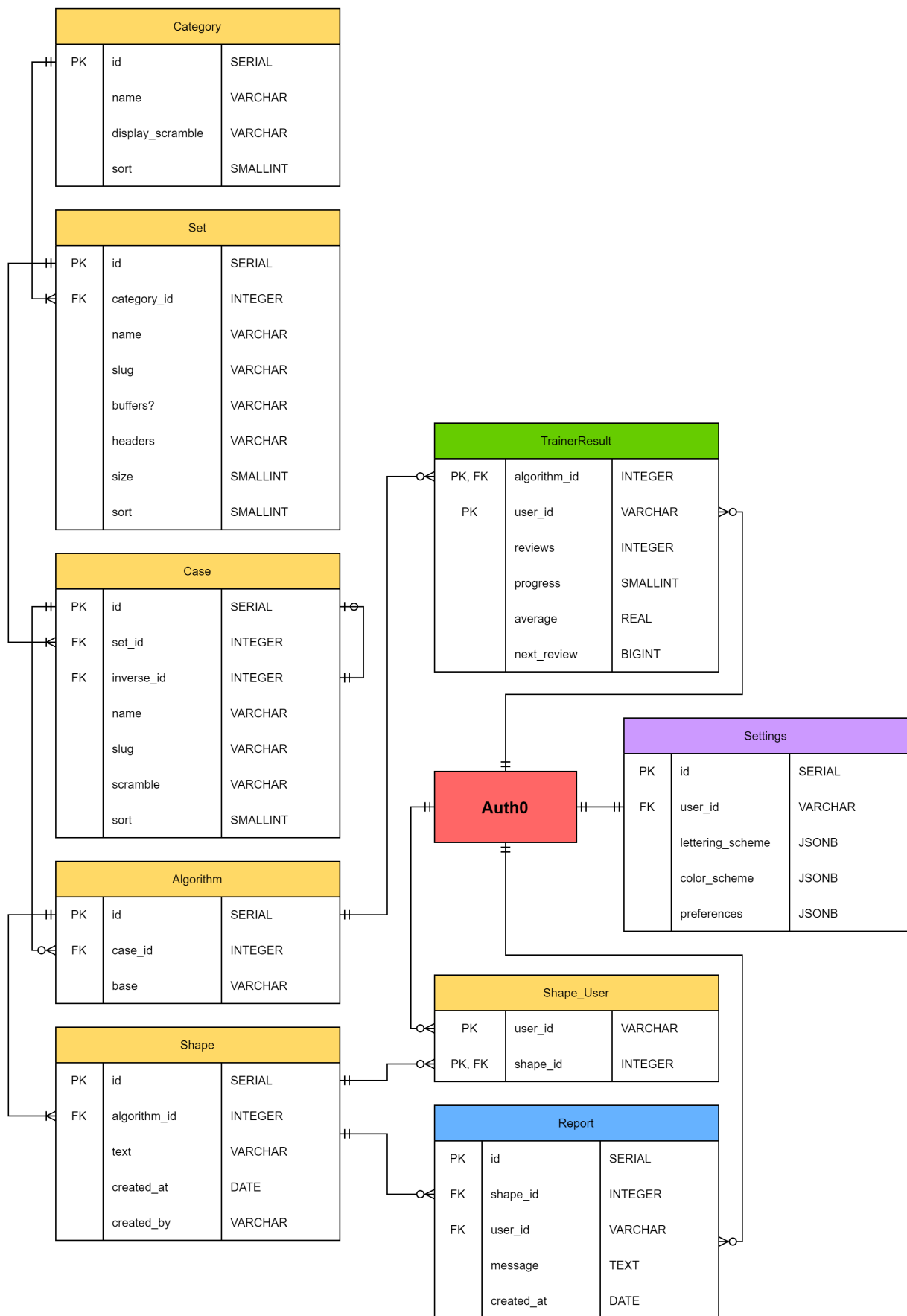
Profile

Review – Corners UFR

KS

Stop

Data model



De tabellen en relaties zijn uitgewerkt op basis van volgende specificaties:

- Sets kunnen gegroepeerd worden per categorie.
- Een set is een verzameling van gerelateerde cases.
- Een case kan opgelost worden door meerdere algoritmes.
- Een algoritme kan op meerdere manieren geschreven worden (shapes).
- Een user gebruikt meerdere algoritmes, en heeft een voorkeur voor een bepaalde shape.
- Een algoritme wordt door meerdere users gebruikt.
- Een user kan een trainer resultaat hebben voor meerdere algoritmes.
- Een algoritme kan een trainer resultaat hebben van meerdere users.
- Iedere user heeft eigen settings.
- Een user kan meerdere algoritmes rapporteren.
- Een algoritme kan door meerdere users gerapporteerd worden.

De `user_id` wordt verkregen van Auth0 als unieke string.

Tech stack

De backend bestaat uit een Express server geschreven in TypeScript. De server gebruikt kysely als query builder, die tevens ook de connectie initialiseert. Winston en morgan worden gebruikt voor logging.

De frontend applicatie is geschreven in React en maakt gebruik van de Vite react-ts template. React Router wordt gebruikt voor client-side navigatie, en React Query voor server state management. Styling gebeurt met vanilla CSS, met een design system gebaseerd op Pollen.

Authenticatie gebeurt via de Auth0 service. Deze wordt geïntegreerd met de frontend via de `auth0-react` package. De server gebruikt `jsonwebtoken` om de JWT's te verifiëren.

Voor de deployment wordt DigitalOcean gebruikt. De applicatie wordt gehost via het App Platform, en de server en database draaien op een droplet. De domeinnaam werd geregistreerd bij Namecheap.

Uitvoering

Library magic-cubes

Aangezien mijn applicatie draait rond Rubik's kubussen, had ik een manier nodig om met kubussen en algoritmes te werken in code. De weinige bestaande libraries hiervoor zijn enorm verouderd en worden niet meer onderhouden, dus ik heb besloten een eigen implementatie te schrijven. Voor de ontwikkeling werd Test Driven Development toegepast met Jest. De grootste uitdaging was het werken met algoritmes. Hiervoor heb ik een lexer en parser geschreven die een bepaalde input omzetten in een AST (Abstract Syntax Tree), zodat deze AST gevalideerd en gemanipuleerd kan worden. Hiervoor zijn enkele visitors geschreven die de AST traversen om zo het algoritme om te kunnen zetten in verschillende vormen. Hoewel de parser enkele belangrijke optimalisaties ontbreekt, is gebleken dat de performance meer dan voldoende is om te gebruiken in de applicatie.

De simulatie van een Rubik's kubus is redelijk eenvoudig. Ik heb gekozen voor een model dat dicht aanleunt bij de realiteit: de kubus bestaat uit acht corners en twaalf edges die elk een bepaalde positie en oriëntatie hebben. Dit staat in contrast met andere uitwerkingen die de kubus simuleren als een (al dan niet geneste) array van 54 kleuren. Deze methode zorgt ervoor dat minder hard-coded transformaties nodig zijn om vlakken in de kubus te draaien. Het bijhouden van de oriëntatie

van de kubus zorgde aanvankelijk voor heel wat problemen. Deze werden uiteindelijk opgelost door de logica apart te verwerken in een eigen klasse. Ook is een klasse voorzien die een kubus kan verwerken en omzetten naar een array van kleuren, zodat de gebruiker van de library eenvoudig een visuele voorstelling kan maken.

De library staat gepubliceerd op npm onder de naam 'magic-cubes' (*Magic Cube* is de originele naam van de Rubik's Cube) en heeft op dit moment versie 0.3.1. Voorlopig ondersteunt de library enkel 3x3 kubussen.

BlindsolveDB

De uitwerking van de applicatie is begonnen met de server en database. Meteen werd een correcte error handling en logging geïmplementeerd om de server robuust te maken in production. *OvernightJS* decorators en *express-promise-router* werden gebruikt om handlers op een meer declaratieve manier te schrijven. Na enkele placeholder routes geschreven te hebben, werd een Auth0 tenant geconfigureerd en werd autorisatie geïmplementeerd in de server.

Hierna werd de database aangemaakt en gevuld met seed data. Express verbindt met de database met behulp van de query builder *kysely*. Er is gekozen voor een query builder omdat ORM's niet de nodige flexibiliteit bieden voor de complexe queries die nodig zijn in deze applicatie. Om bepaalde joins niet telkens te moeten herhalen voor het ophalen van data, werden vijf views aangemaakt om deze queries te vereenvoudigen. Ook werden enkele indexen toegevoegd aan de database om de leessnelheid te verhogen. *Kysely* wordt ook gebruikt om met migraties te werken. Aangezien *kysely* nog geen stabiele release heeft, wordt de development nauw opgevolgd via Github en Discord.

Met werkende controllers en een werkende database connectie, werden de route handlers uitgewerkt. Na het schrijven van de handlers voor de sheets, is begonnen aan de development van de frontend. De andere features werden gelijktijdig uitgewerkt in front- en backend.

Voor de frontend werd vertrokken vanuit de React-TypeScript template van *Vite*. Na een minimale opstelling gemaakt te hebben met *React Router* en *React Query*, werd een design system uitgewerkt. Hiervoor werd CSS-in-JS en CSS Modules overwogen, maar werd uiteindelijk gekozen voor vanille CSS, met enkele PostCSS plugins voor minification en auto-prefixing. Met de belangrijkste components bij de hand, werden de sheets pagina's geïmplementeerd. Op dit moment kwamen enkele problemen naar boven in verband met sets die onderveeld konden worden in subsets. De beste oplossing bleek uiteindelijk het verwijderen van deze relatie, wat betekende dat de many-to-many relatie tussen de *Set* en *Case* tabel verwijderd werd. Dit vereenvoudigde het data model en liet het ontwikkelen van enkele nieuwe features toe.

Hierna werd Auth0 geïntegreerd om vervolgens de user-specifieke pagina's te ontwikkelen. Dit werd gevolgd door het uitwerken van de trainer, de profielpagina en het admin dashboard. Wanneer nodig werden nieuwe design components aangemaakt, al dan niet met behulp van enkele hooks van *React Aria*.

Na het uitwerken van alle features, werden een groot aantal QOL aanpassingen gemaakt, gevolgd door de deployment van de applicatie. Voor de deployment van de frontend werd gekozen voor DigitalOcean's App Platform, aangezien deze optie gratis is voor statische websites en de bronbestanden worden verdeeld over hun CDN. De backend en database worden gehost op een DigitalOcean Droplet, om de prijs laag te houden in vergelijking met aparte hosting van de server en database. Een extra volume werd gekoppeld aan deze VPS om backups van de database op te slaan.

BESLUIT

Conclusies

In het kader van dit onderzoek werd een literatuuronderzoek uitgevoerd om antwoord te kunnen bieden op de onderzoeksvraag. Er werd gezocht naar de struikelpunten bij het leren van blindsolving. Uit de resultaten bleek dat 37% van de cubing community wilt leren blindsolven maar dat er een tool ontbreekt dat het leerproces gemakkelijker en tijdsefficiënt maakt. Na een grondige concurrentieanalyse, werd de data gebruikt om de webapplicatie BlindsolveDB te ontwikkelen. BlindsolveDB biedt de gebruiker de voordelen van spreadsheets, gecombineerd met de features van databases als algdb.net. De ingebouwde trainer staat toe om algoritmes te leren in kleine stukken, en gebruikt *spaced repetition* om te weten wanneer algoritmes herhaald moeten worden, zodat het leerproces zo optimaal mogelijk kan verlopen. Met de verschillende personalisatieopties kan iedere gebruiker kiezen voor een persoonlijke user experience, afgestemd op zijn of haar manier van blindsolven. Omdat er nood was om in de applicatie te werken met Rubik's kubussen en algoritmes, werd de library 'magic-cubes' geschreven. In de toekomst zal de applicatie support bieden voor 4BLD en 5BLD, aangezien op dit moment enkel 3x3 algoritmes ondersteund worden.

Reflectie

Ondanks dat ik zelf enkele jaren ervaring heb met speedcubing, heeft het onderzoek naar het onderwerp mij op enkele vlakken kunnen helpen. Door mij te verdiepen in het onderwerp, ben ik tot bepaalde inzichten gekomen die ik ervoor niet had. Voor het doelgroeponderzoek had ik het geluk de Mega Survey's van de Reddit speedcubing community te kunnen gebruiken. Hierin was veel informatie te vinden die ik nodig had voor mijn onderzoek. Verrassend vond ik dat de interesse in het leren van blindsolving vrij hoog lag. Ook heeft de concurrentieanalyse mij veel geholpen bij het uitwerken van mijn werkstuk. Ik heb eruit geleerd welke features handig kunnen zijn en welke overbodig.

Het werkstuk lijkt mij een goed antwoord te zijn op de problemen die ik met het onderzoek ontdekt heb. Nu is het uiteraard belangrijk de applicatie te testen met de doelgroep, aangezien hiervoor geen tijd meer was. Ik denk dat ik aan de belangrijkste noden heb voldaan, maar verwacht wel dat enkele zaken bijgewerkt zullen moeten worden op basis van de feedback.

REFERENTIELIJST

- *3x3x3 notation*. (z.d.). Speedsolving.Com Wiki. Geraadpleegd op 28 januari 2022, van https://www.speedsolving.com/wiki/index.php/3x3x3_notation
- *AlgDb*. (z.d.). AlgDb. Geraadpleegd op 21 januari 2022, van <http://algdb.net>
- *Algdb.net*. (z.d.). Speedsolving.Com Wiki. Geraadpleegd op 21 januari 2022, van <https://www.speedsolving.com/wiki/index.php/Algdb.net>
- *Anki - powerful, intelligent flashcards*. (z.d.). Anki. Geraadpleegd op 23 januari 2022, van <https://apps.ankiweb.net>
- *Beyer-Hardwick Method*. (z.d.). SpeedSolving.Com Wiki. Geraadpleegd op 18 januari 2022, van https://www.speedsolving.com/wiki/index.php/Beyer-Hardwick_Method
- Cai, J. (2021, 2 juli). *#1 Underrated Tool For Blindsolving Improvement (Part 3/6)*. YouTube. Geraadpleegd op 23 januari 2022, van <https://www.youtube.com/watch?v=UtanRYW54vk>
- *Commutator*. (z.d.). Speedsolving.Com Wiki. Geraadpleegd op 28 januari 2022, van <https://www.speedsolving.com/wiki/index.php/Commutator>
- *Commutators and Conjugates*. (z.d.). Speedsolving.Com Wiki. Geraadpleegd op 28 januari 2022, van https://www.speedsolving.com/wiki/index.php/Commutators_and_Conjugates
- *Conjugate*. (z.d.). Speedsolving.Com Wiki. Geraadpleegd op 28 januari 2022, van <https://www.speedsolving.com/wiki/index.php/Conjugate>
- *History of the Speedcubing Community & WCA*. (z.d.). World Cube Association. Geraadpleegd op 4 december 2021, van <https://www.worldcubeassociation.org/speedcubing-history>
- *M2/R2*. (z.d.). Speedsolving.Com Wiki. Geraadpleegd op 18 januari 2022, van <https://www.speedsolving.com/wiki/index.php/M2/R2>
- McNeill, J. (2020, 5 augustus). *How to kill algdb*. YouTube. Geraadpleegd op 21 januari 2022, van <https://www.youtube.com/watch?v=oc28sO4Ulos>
- Noris, B. (2021, 26 juli). *The Cubing Community Megasurvey 2021*. Basilio. Geraadpleegd op 3 november 2021, van <https://basilio.dev/cubing/megasurvey/CubingMegasurvey2021.pdf>
- */r/Cubers Mega-Survey 2 Results and Analysis (Preliminary)*. (z.d.). Google Docs. Geraadpleegd op 3 november 2021, van <https://docs.google.com/document/d/1wzv1ijxh7QP86X5C7zXsS18up0MvlgPWqniPsSGFZo/edit>
- */r/Cubers Mega-Survey 3 Preliminary Results*. (z.d.). Google Docs. Geraadpleegd op 3 november 2021, van <https://docs.google.com/document/d/1rMqshx4UFSTQKykzOffvrjQwI3GJ-eO42UwC5XjHcEo/edit>
- */r/Cubers Mega-Survey 4 Summary*. (z.d.). Google Docs. Geraadpleegd op 3 november 2021, van https://docs.google.com/document/d/14qWyxxYjKMOv1v0-fRR0lcdzJG8Eyfgldvji_cMEXic/edit
- */r/Cubers Mega-Survey Results and Analysis (part 1)*. (z.d.). Google Docs. Geraadpleegd op 3 november 2021, van https://docs.google.com/document/d/1k1YNnZDiHjvZmu1poX0pQxdDfYil674PIkilmD_qfsk/edit

- *Records*. (z.d.). World Cube Association. Geraadpleegd op 18 januari 2022, van <https://www.worldcubeassociation.org/results/records>
- *Speedcubedb*. (z.d.). Speedsolving.Com Wiki. Geraadpleegd op 21 januari 2022, van <https://www.speedsolving.com/wiki/index.php/Speedcubedb>
- *Speffz*. (z.d.). Speedsolving.Com Wiki. Geraadpleegd op 16 januari 2022, van <https://www.speedsolving.com/wiki/index.php/Speffz>
- *Unified BLD Algorithm Database*. (z.d.). Unified BLD Algorithm Database. Geraadpleegd op 19 januari 2022, van <https://bldbase.net>
- Wang, D. (z.d.). *J Perm / Speedcubing Tutorials*. J Perm. Geraadpleegd op 22 januari 2022, van <https://jperm.net>
- *WCA Regulations*. (z.d.). World Cube Association. Geraadpleegd op 5 december 2021, van <https://www.worldcubeassociation.org/regulations>
- Wikipedia contributors. (2022a, april 9). *World Cube Association*. Wikipedia. Geraadpleegd op 4 december 2021, van https://en.wikipedia.org/wiki/World_Cube_Association
- Wikipedia contributors. (2022b, april 10). *Method of loci*. Wikipedia. Geraadpleegd op 16 januari 2022, van https://en.wikipedia.org/wiki/Method_of_loci
- Wikipedia contributors. (2022c, april 14). *Speedcubing*. Wikipedia. Geraadpleegd op 4 december 2021, van <https://en.wikipedia.org/wiki/Speedcubing>
- Wikipedia contributors. (2022d, april 15). *Rubik's Cube*. Wikipedia. Geraadpleegd op 28 januari 2022, van https://en.wikipedia.org/wiki/Rubik%27s_Cube
- World Cube Association. (z.d.). *WCA results export*. Geraadpleegd op 14 november 2021, van <https://www.worldcubeassociation.org/results/misc/export.html>
- Zussman, G. (z.d.). *Speed Cube Database*. SpeedCubeDB. Geraadpleegd op 21 januari 2022, van <https://www.speedcubedb.com>

BIJLAGES

Bijlage 1: WCA events en formats

Event	Format
3x3x3	ao5
2x2x2	ao5
4x4x4	ao5
5x5x5	ao5
6x6x6	mo3
7x7x7	mo3
3x3x3 Blindfolded (3BLD)	bo3
3x3x3 Fewest Moves (FMC)	bo1/bo2/mo3
3x3x3 One-Handed (OH)	ao5
Megaminx	ao5
Pyraminx	ao5
Clock	ao5
Skewb	ao5
Square-1	ao5
4x4x4 Blindfolded (4BLD)	bo3
5x5x5 Blindfolded (5BLD)	bo3
3x3x3 Multi-Blind (MBLD)	bo1/bo2/bo3

In het merendeel van de events wordt het *average of 5* (ao5) format gebruikt. Dit wil zeggen dat zowel de snelste als de traagste tijd van vijf pogingen genegeerd worden, en het gemiddelde van de overige drie tijden berekend wordt om het eindresultaat te bepalen. Bij 6x6, 7x7 en FMC wordt het *mean of 3* (mo3) format gebruikt, waarbij slechts 3 solves uitgevoerd worden en van deze drie het gemiddelde berekend wordt. De blindfolded events gebruiken het *best of 3* (bo3) format: enkel de snelste tijd van de drie resultaten is hier van belang. Om tijd te besparen kunnen competities er ook voor kiezen om slechts één of twee solves te doen per ronde. Dit wordt gedaan bij FMC of MBLD, waarbij dan het bo1 of bo2 format gebruikt wordt.

Hoewel naar het format gekeken wordt om de tussenstand te bepalen op een competitie, worden zowel de snelste tijd (single) als de gemiddelde tijd (ao5 of mo3) bijgehouden bij records. Yusheng Du heeft bijvoorbeeld het wereldrecord van snelste single 3x3x3 met een tijd van 3,47 seconden behaald, maar won hiermee de competitie niet omdat hij niet de snelste ao5 tijd had.

Bijlage 2: Speedsolve verloop

Een speedsolve poging bestaat uit een aantal stappen, welke beschreven staan in Artikel A van het WCA reglement. Bij aanvang van een ronde, dienen de deelnemers hun puzzels in. Deze worden door elkaar gehaald aan de hand van een computer-gegenereerde *scramble*. Iedere deelnemer zal dezelfde scrambles krijgen in een ronde. Wanneer de puzzel door elkaar gehaald is, wordt de deelnemer opgeroepen voor zijn poging. De deelnemer neemt plaats aan een tafel waarop een timer voorzien is, en een jurylid de poging zal overzien. De puzzel wordt bedekt op tafel geplaatst. Wanneer de deelnemer klaar is, wordt de puzzel onthuld en begint de inspectiefase. Tijdens de inspectie mag de puzzel voor maximum 15 seconden bekeken worden alvorens de timer gestart wordt. Na de inspectietijd plaatst de speler beide handen op de timer en begint de solve. De solve eindigt wanneer de speler de puzzel loslaat en terug beide handen op de timer plaatst. Het jurylid schrijft het resultaat neer op het scoreblad dat door zowel het jurylid als de deelnemer ondertekend wordt.



Chris Tran tijdens een 3x3 One-Handed solve. Voor hem op tafel is de timer te zien, die verbonden is met een display. Wat verder staat de cube cover die gebruikt werd om de puzzel af te dekken, en de chronometer die het jurylid gebruikt heeft om de inspectietijd te timen. (Foto van wbur.org)

Afhankelijk van het event, bestaat een ronde uit drie of vijf solves. Als de puzzel niet opgelost is wanneer de timer gestopt wordt, krijgt de deelnemer een DNF (did not finish) als resultaat. Bij eventuele overtredingen kunnen strafseconden toegevoegd worden aan de tijd (+2) of wordt een DNF toegekend.

Bijlage 3: Rubik's cube turns



U



D



R



L



F



B



Uw / u



Dw / d



Rw / r



Lw / l



Fw / f



Bw / b



x



y



z



M



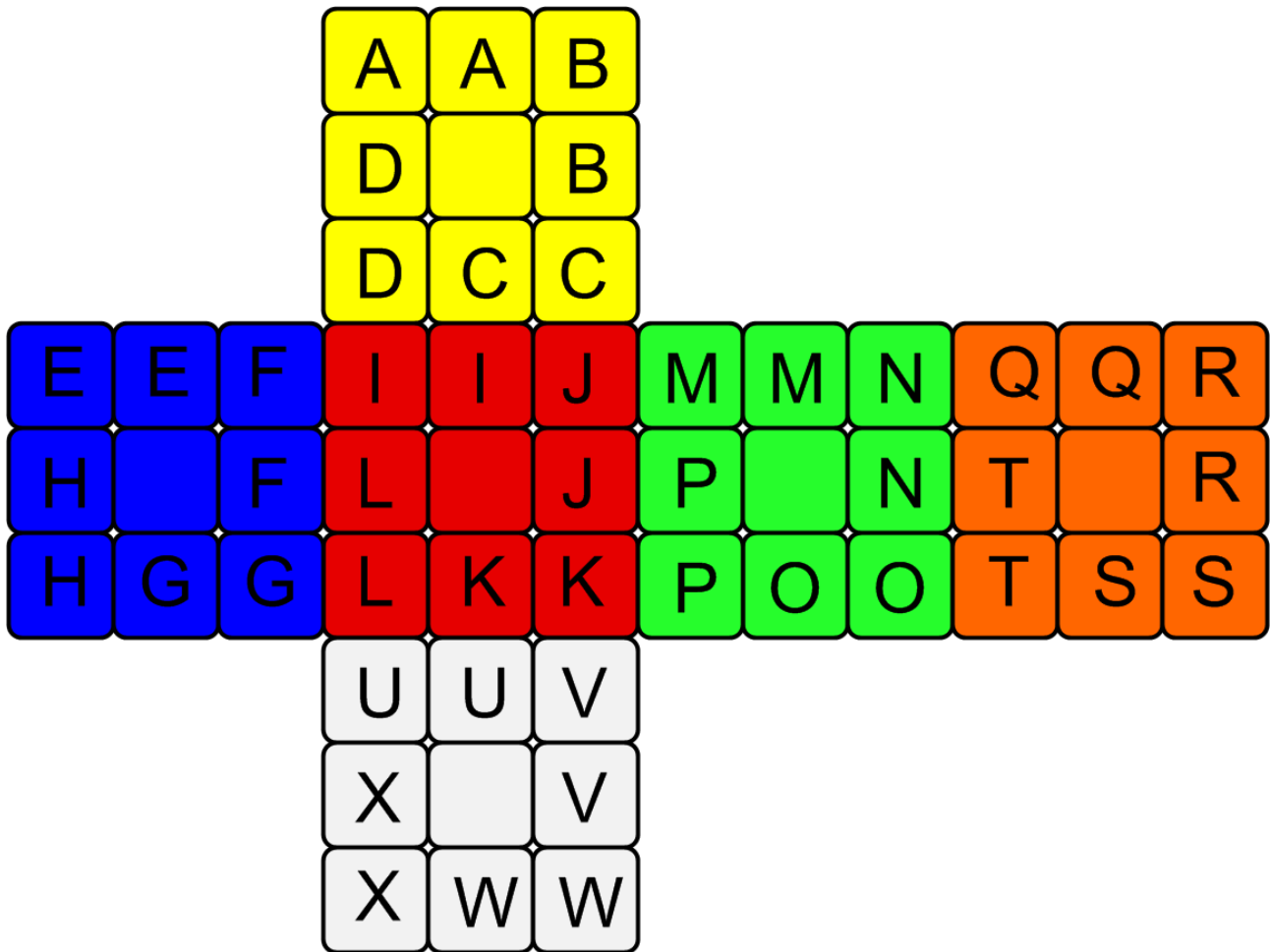
E



S

Afbeelding van jperm.net

Bijlage 4: Speffz lettering scheme



Afbeelding van teachkidsengineering.com

Bijlage 5: Memorisatiemethodes

De verhalenmethode houdt in dat van elk letterpaar een woord gevormd wordt, en dat de woorden samen een beeld of verhaal vormen. Het voordeel van deze methode is dat de verbeelding gebruikt wordt, waardoor de memo voor een langere tijd onthouden kan worden. Het nadeel is echter dat voor elk letterpaar minstens één woord gekend moet zijn dat visueel voor te stellen is, wat niet voor elke combinatie van letters evident is. Sommige blindsolvers stellen een lijst op van alle letterparen en hun bijbehorende woorden, om te vermijden dat nieuwe woorden tijdens een solve bedacht moeten worden. Een uitbreiding op deze methode is het Person-Action-Object (PAO) systeem, waarin elk woord afwisselend een persoon, een actie en een voorwerp is. Deze sequentie van woorden maakt het creëren van zinnen veel eenvoudiger en natuurlijker, maar dit wil wel zeggen dat voor elk letterpaar drie woorden gekend moeten zijn.

De audio methode bestaat uit het memoriseren van klanken die gemaakt worden op basis van de reeks letterparen. Deze methode staat toe om heel snel te memoriseren, maar men kan de letterreeks minder lang onthouden waardoor deze gevoeliger is voor fouten.

Visuele memorisatie houdt in dat op basis van ruimtelijk inzicht de eindposities van de individuele cubies onthouden worden. Hierbij wordt geen gebruik gemaakt van letters om te posities aan te duiden. Wegens de moeilijkheid van deze methode wordt ze weinig toegepast voor een volledige memo, maar vaak wordt visuele memorisatie aanvullend gebruikt bij andere methodes.

De loci-methode, ook wel de plaatsmethode of *memory palace* genoemd, is een geavanceerde methode voor het memoriseren van grote hoeveelheden informatie. Het vindt zijn oorsprong bij de oude Grieken en Romeinen, die het gebruikten voor het onthouden van lange redevoeringen. Hierbij wordt mentaal door een ruimte gewandeld en worden op verschillende plekken in deze ingebeelde ruimte 'objecten' neer geplaatst. Deze 'objecten' zijn vaak beelden zoals in de verhalenmethode en kan gecombineerd worden met de PAO-techniek. Na het plaatsen van de memo in de ruimte, kan de mentale wandeling opnieuw gemaakt worden en kan er rondgekeken worden naar de memo die zich in de ruimte bevindt. Door het linken van de memo met een gekende plaats, kan de memo voor een erg lange tijd onthouden worden. De opstelling duurt echter langer dan de andere methodes en is vaak overbodig voor een snelle memorisatie. De loci-methode wordt voornamelijk gebruikt in MBLD, 5BLD en in mindere mate 4BLD.