2015•2016
FACULTEIT BEDRIJFSECONOMISCHE WETENSCHAPPEN
*master in de toegepaste economische wetenschappen:*
*handelsingenieur in de beleidsinformatica*

# Masterproef

## Een analyse van de evaluatiemetrieken voor process discovery

Promotor :
Prof. dr. Benoit DEPAIRE

Copromotor :
De heer Gert JANSSENSWILLEN

Niels Donders

*Scriptie ingediend tot het behalen van de graad van master in de toegepaste economische wetenschappen: handelsingenieur in de beleidsinformatica*

universiteit hasselt
KNOWLEDGE IN ACTION

Universiteit Hasselt | Campus Hasselt | Martelarenlaan 42 | BE-3500 Hasselt
Universiteit Hasselt | Campus Diepenbeek | Agoralaan Gebouw D | BE-3590 Diepenbeek

universiteit hasselt
KNOWLEDGE IN ACTION

2015•2016
FACULTEIT BEDRIJFSECONOMISCHE WETENSCHAPPEN
*master in de toegepaste economische wetenschappen: handelsingenieur in de beleidsinformatica*

# Masterproef

## Een analyse van de evaluatiemetrieken voor process discovery

Promotor :
Prof. dr. Benoit DEPAIRE

Copromotor :
De heer Gert JANSSENSWILLEN

Niels Donders

*Scriptie ingediend tot het behalen van de graad van master in de toegepaste economische wetenschappen: handelsingenieur in de beleidsinformatica*

universiteit hasselt
KNOWLEDGE IN ACTION

# Woord vooraf

Deze masterproef vormt het sluitstuk van de opleiding Toegepaste Economische Wetenschappen - Handelsingenieur in de Beleidsinformatica aan de universiteit Hasselt. Het onderwerp van deze thesis handelt over een analyse van de evaluatiemetrieken voor process discovery. De keuze voor dit onderwerp was relatief snel gemaakt, gezien de aanwezige interesse voor zowel process mining als datanalyse. Deze masterproef zag ik dan ook als een uitdaging om mij verder te verdiepen in beide onderwerpen.

Ondanks de interesse, was deze masterproef niet tot stand gekomen zonder de, al dan niet bewuste, hulp en ondersteuning van tal van personen. In de eerste plaats wil ik uiteraard mijn promotor prof. dr. Benoit Depaire bedanken, niet enkel voor de constructieve feedback en begeleiding, maar ook voor zijn interessante en boeiende lessen die mede de interesse in bovenstaande vakgebieden hebben aangewakkerd. Verder wil ik mijn copromotor Gert Janssenswillen en Toon Jouck bedanken omdat ik bij hen steeds terecht kon met vragen. Het was zeer aangenaam om met hen te mogen samenwerken en hun nuttige adviezen hebben geholpen om de kwaliteit van de masterproef te verbeteren.

Ten slotte wil ik graag mijn naaste familie en vrienden bedanken voor hun morele steun. In het bijzonder mijn ouders, voor de kansen die ze mij gegeven hebben tijdens deze opleiding. Ook mijn vriendin wil ik bedanken voor de vele woorden van steun en geloof in mij. Zonder hen, was deze thesis niet geworden wat het vandaag is.

Niels Donders

Zonhoven, mei 2016

# Samenvatting

De hoeveelheid data waarover bedrijven vandaag de dag kunnen beschikken is immens. Ook in verband met hun eigen bedrijfsprocessen, kan zeer veel data verzameld worden. Heden ten dage zijn er tal van informatiesystemen die voortdurend data opslaan over de activiteiten en transacties die in een bedrijf plaatsvinden. Deze data worden opgeslagen in event logs. Bedrijven kunnen aan de hand van verschillende technieken deze event logs analyseren om er procesmodellen uit te ontdekken. Deze tak van process mining heet process discovery. Nadat de processen zijn ontwikkeld, kan nagegaan worden of ze overeenkomen met de werkelijke processen in het bedrijf om vervolgens eventuele verbeteringen toe te passen, wat conformance checking en enhancement heet.

Om de overeenkomst van een event log met het respectievelijke procesmodel te meten, werden het afgelopen decennium verschillende metrieken ontwikkeld die de accuraatheid (precision, fitness en generalization) en de complexiteit (simplicity en structuredness) kwantificeren. Nochtans is het uit de literatuur onduidelijk wat werkelijk gemeten wordt door elke metriek en hoe ze zich ten opzichte van elkaar verhouden. Er zijn veel verschillende metrieken voor elke dimensie beschikbaar die elk op een andere manier hun respectievelijke dimensie trachten te kwantificeren. Er zal daarom eerst een grondige literatuurstudie uitgevoerd worden van elke metriek die daarna verder geanalyseerd zal worden. Uit deze literatuurstudie wordt duidelijk dat het moeilijk te voorspellen is hoe de metrieken gaan reageren op een gegeven situatie, doordat elke metriek een andere methode toepast. Bovendien is het onduidelijk of de verschillende metrieken in de praktijk wel degelijk de dimensie meten die ze theoretisch beweren te meten. Het is tenslotte niet duidelijk hoe de metrieken zich ten opzichte van elkaar verhouden.

Naast deze bevindingen werd er eveneens een contradictie gevonden omtrent de onafhankelijkheid tussen de dimensies fitness en precision. Deze onafhankelijkheid blijkt namelijk in enkele van de precision-metrieken niet voldaan te zijn. Deze precision-metrieken gaan, alvorens hun berekening te starten, de traces aligneren met het model zodat alle traces in rekening worden genomen, ook de non-fitting traces. Bij een lage fitness, als er dus veel non-fitting cases zijn, zullen deze alignment-based metrieken een positiever beeld vormen van de precision dan het in werkelijkheid is.

Om zowel meer duidelijkheid te scheppen over de verschillende metrieken, als de gevonden contradictie te onderzoeken worden er enkele experimenten opgezet met artificiële events logs. Deze event logs werden verkregen door uit een populatie van processmodellen een steekproef te trekken. Vervolgens genereerde een simulator event logs met verschillende niveaus van completeness en noise uit elk van de procesmodellen in de steekproef. Vervolgens worden deze event logs gebruikt om, aan de hand van verschillende process discovery technieken, modellen te ontdekken. Hiermee kunnen dan de verschillende gekozen metrieken berekend en geanalyseerd worden. Er wordt gekozen om gebruik te maken van verschillende types van analyses, waaronder een correlatie- en factoranalyse voor het eerste experiment, en een regressieanalyse voor het tweede experiment. Deze analyses zullen meer inzicht geven in de gevonden onduidelijkheden of tegenstrijdigheden.

Uit de analyse van het eerste experiment blijkt dat de verschillende fitness- en precision-metrieken hun respectievelijke dimensie op een degelijke manier kwantificeren. Voor de dimensie precision gedragen de One Align, Best Align als de Behavioral Precision zich zeer gelijkaardig in geval van zowel eenvoudigere als meer complexe modellen. De Token Based Fitness en de Behavioral Recall, die beiden de dimensie fitness kwantificeren, gedragen zich eveneens gelijkaardig. Nochtans hebben sommige van deze metrieken meer rekencapaciteit nodig dan andere. De One Align en Best Align Precision hebben meer moeite met het komen tot een oplossing in geval van meer complexe modellen, terwijl de Behavioral Precision dit niet heeft. Ook de Token Based Fitness heeft meer capaciteit nodig dan de Behavioral Recall, maar in mindere mate dan de One Align en Best Align Precision.

Naast deze metrieken worden ook de Alignment Based metrieken (Alignment Based Fitness en Precision) opgenomen in de analyse. Ook deze metrieken hebben zeer veel moeite met de berekening als er complexe modellen gebruikt worden. Bovendien gaan deze metrieken zich verschillend gedragen in geval van complexe modellen, vergeleken met de andere metrieken binnen hun respectievelijke dimensies. Deze metrieken zijn in dat geval minder betrouwbaar ten opzichte van de andere metrieken. Als er echter eenvoudige modellen van hogere kwaliteit gebruikt worden (in geval van hogere fitness en precision), blijken deze metrieken wel gebruikt te kunnen worden voor een betrouwbare interpretatie van de dimensies fitness en precision.

Niet enkel metrieken voor de dimensies fitness en precision werden opgenomen in de analyse, ook de Behavioral en Alignment Based Generalization worden geanalyseerd. Nochtans blijkt uit deze analyse dat beide metrieken niet de dimensie generalization kwantificeren. De Behavioral Generalization blijkt zelfs eerder de dimensie fitness weer te geven. Deze metriek correleert namelijk relatief sterk met de andere metrieken die deze dimensie kwantificeren. Over de Alignment Based Generalization kan ten slotte moeilijk een conclusie getrokken worden, aangezien deze metriek met geen enkele andere metriek in het experiment correleert.

Verder blijkt dat in geval van eenvoudige modellen, de fitness en precision negatief gecorreleerd zijn aan elkaar. Nochtans, als de modellen meer complex worden en bijgevolg realistischer zijn, verdwijnt deze correlatie. Enkel de Alignment Based Precision is dan nog licht negatief gecorreleerd aan de fitness-metrieken.

In het algemeen valt op dat elke metriek anders reageert op een veranderend niveau van kwaliteit van het gebruikte model. Als de fitness hoog is, liggen de verschillende fitness-metrieken relatief kort bij elkaar. Naarmate de fitness daalt, wordt de spreiding tussen deze metrieken groter. De Behavioral Recall kwantificeert de fitness in het algemeen lager dan de Token Based Fitness. De Alignment Based Fitness kwantificeert in geval van een hogere kwaliteit van het model, de fitness doorgaans lager dan de andere twee fitness-metrieken. In het geval van een lagere kwaliteit van het model gaat deze metriek de fitness echter positiever kwantificeren. De Alignment Based Precision kwantificeert de dimensie precision in het algemeen hoger dan de andere precision-metrieken, terwijl de Behavioral Precision deze dimensie doorgaans lager kwantificeert. De One Align en Best Align Precision liggen over het gehele bereik kort bij elkaar. Enkel bij een zeer hoge kwaliteit van het

model, wijken deze metrieken van elkaar en gaat de Best Align Precision de precision lager kwantificeren.

In het tweede experiment werden statistisch significante invloeden gevonden van het aantal non-fitting traces op de One Align en Behavioral Precision, respectievelijk een positieve en negatieve invloed. Er werd dus geen invloed gevonden op de Best Align en Alignment Based Precision. Nochtans is dit resultaat een aanduiding dat de onafhankelijkheid tussen de fitness en precision niet in elke metriek gehandhaafd wordt. Naast de precision-metrieken, werd ook de invloed van het niveau van non-fitting traces op de andere metrieken onderzocht. Zo bleek dat er een negatieve significante invloed aanwezig is op zowel de Token Based en Alignment Based Fitness als de Behavioral Recall. Dit resultaat werd verwacht, ook al is het effect veel minder sterk dan oorspronkelijk gedacht. Dit wijst erop dat de geobserveerde fitness-metrieken deze dimensie positiever weergeven dan de werkelijkheid. Voor beide geobserveerde generalization-metrieken werd eveneens een significante invloed gevonden. Voor de Alignment Based Generalization was deze invloed slechts gering, terwijl de Behavioral Generalization zeer gevoelig is aan het niveau van non-fitting cases. Dit wijst erop dat, net zoals in het eerste experiment, deze metriek eerder de dimensie fitness kwantificeert.

Op basis van het uitgevoerde onderzoek wordt er aanbevolen om gebruik te maken van de Token Based Fitness of de Behavioral Recall. Rekening houdende met de benodigde rekencapaciteit die beide metrieken nodig hebben, blijkt de Behavioral Recall nochtans de beste optie te zijn. Om de dimensie precision te kwantificeren zijn de One Align en de Behavioral Precision het meest aangewezen. Opnieuw rekening houdende met de benodigde rekencapaciteit, kan best de Behavioral Precision gebruikt worden. Voor de dimensie generalization werden er echter geen betrouwbare metrieken gevonden. Beide geobserveerde metrieken waren niet in staat deze dimensie correct weer te geven, waardoor er verder onderzoek verricht moet worden.

# Inhoudsopgave

# Hoofdstuk 1: Probleemstelling en onderzoeksaanpak

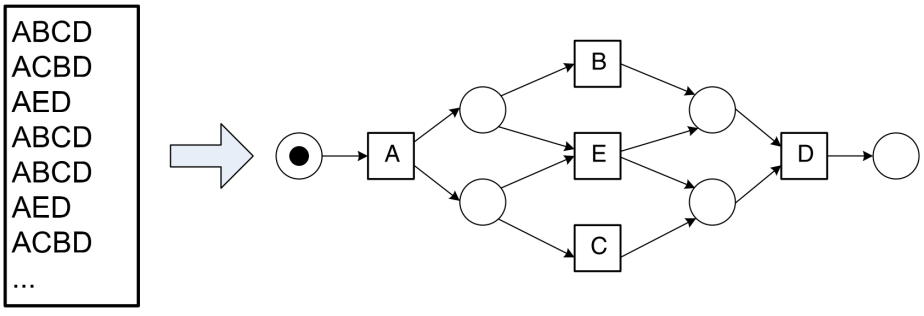
## 1.1 Inleiding

De hoeveelheid data waarover bedrijven vandaag de dag beschikken is immens. Niet alleen data over hun klanten, leveranciers of verkopen, maar ook zeer veel data in verband met hun eigen bedrijfsprocessen kan verzameld worden met behulp van transactionele informatiesystemen (bv. WFM, ERP, CRM, SCM en B2B systemen) [1]. Er worden zoveel mogelijk data opgeslagen over hoe de werknemer de bedrijfsprocessen toepast en welk proces er op welk moment plaatsvindt. Deze procesdata worden event logs genoemd. Deze logs bevatten gewoonlijk informatie die verwijzen naar de activiteit en de case. De case is hetgeen behandeld wordt, bijvoorbeeld een klantenorder of een verzekeringsclaim. De activiteit (taak, operatie, actie of work-item) is de operatie die op de case wordt toegepast. Gewoonlijk bevatten de event logs ook een timestamp en een actor (een persoon, resource of systeem component) die het event uitvoert [1], [2].

Tot een tiental jaar geleden werden deze logs nog zeer weinig gebruikt om analyses op toe te passen, maar het afgelopen decennium begonnen bedrijven er steeds meer aandacht aan te besteden. Het verkrijgen van inzichten in en het optimaliseren van de bedrijfsprocessen kan namelijk een (immens) bedrijfseconomisch voordeel opleveren. [3]

Het domein dat zich bezighoudt met knowledge discovery uit event logs heet process mining. Het is een onderzoeksgebied dat zich situeert binnen zowel data mining als Business Process Management (BPM). Er zijn drie types van process mining: discovery, conformance en enhancement [4].

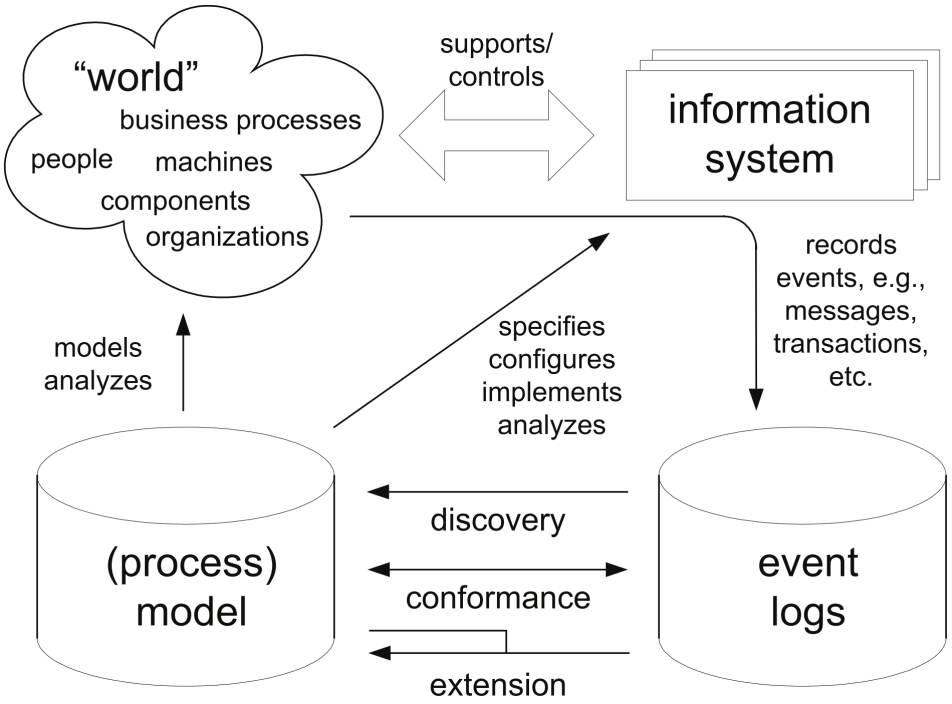
Discovery is het produceren van een model, gegeven een event log, zonder enige andere a priori data te gebruiken [2], [4]. Er bestaat nochtans veel onduidelijkheid in de literatuur over het verschil tussen process mining en process discovery. Vandaar volgend verklarend citaat uit [5]: "Process mining describes a family of a posteriori analysis techniques for extracting knowledge from event logs, whereas process discovery only deals with extracting control-flow models." Process discovery houdt zich dus enkel bezig met het opstellen van de workflow modellen op basis van een gegeven event log, terwijl process mining in het algemeen kennis probeert te vergaren uit de event logs [4].




*Figuur 1: Een event log en het procesmodel dat er uit ontdekt wordt*

Na het ontwikkelen van deze procesmodellen met behulp van process discovery, kan er door het bedrijf nagegaan worden hoe de werknemers en/of processen transacties uitvoeren in de realiteit. Het krijgt met andere woorden een zicht op de workflows die in de realiteit gehanteerd worden. De

verschillende informatiesystemen houden gegevens bij van elke transactie in elk proces van elke werknemer die er gebruik van maakt, maar het verplicht in de meeste gevallen een werknemer niet om op een bepaalde manier te werken. Vervolgens kan het ontdekte model vergeleken worden met een model dat op voorhand werd opgesteld. Dit laatste model geeft weer hoe het proces er zou moeten uitzien. De zoektocht naar overeenkomsten of verschillen tussen deze twee modellen wordt conformance checking genoemd. Er wordt met andere woorden gecontroleerd of de realiteit, zoals opgenomen in de event log, conform het model is en omgekeerd [2], [4], [6]. Andere namen hiervoor zijn Delta analysis en business alignment [1], [6], [7]. De vraag die gesteld wordt is of de reële processen en de procesmodellen goed gealigneerd zijn. Deze masterproef zal zich voornamelijk toespitsen op conformance checking. Figuur 2 geeft weer waar conformance zich bevindt binnen het vakgebied van process mining [8].




*Figuur 2: Situering process mining [8]*

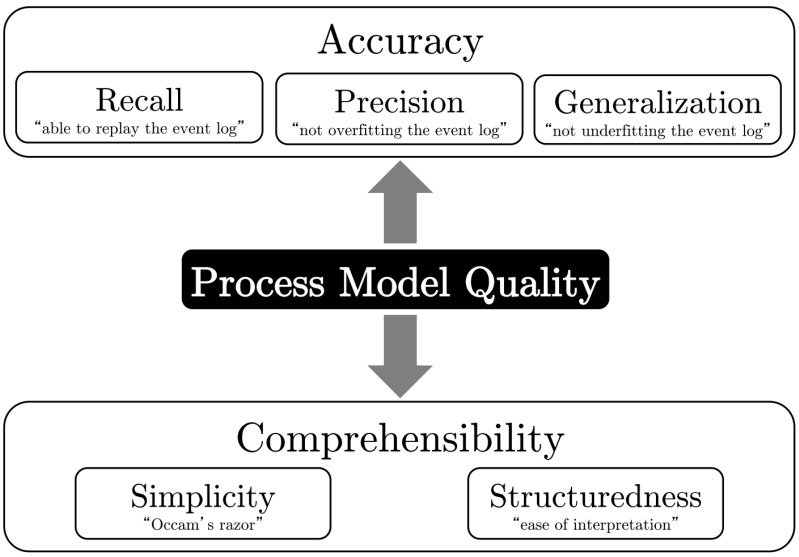
Met behulp van process mining kunnen dus inzichten verworven worden in het werkelijke proces en ontstaat ook een soort controlemiddel om de vooropgestelde werking van het bedrijf te vergelijken met de realiteit [1], [7].

Tot slot wordt er bij enhancement een event log en een procesmodel genomen om dit model vervolgens uit te breiden (extension) of te verbeteren, door middel van de geobserveerde events [2], [4]. Aan de hand van de conclusies die men trekt bij process discovery en conformance kunnen beslissingen genomen worden voor de verbetering van het proces. Dit wordt ook wel Business Process Reengineering (BPR) genoemd [7].

Voor elk van deze domeinen van process discovery zijn er ettelijke technieken, frameworks en/of programma's ontwikkeld die de gebruiker ondersteunen in het proces. Zo zijn er het afgelopen decennium verschillende process discovery technieken ontwikkeld om event logs te analyseren en

hieruit business processen te ontwikkelen. Voorbeelden van deze technieken zijn onder meer het α-algoritme, $\alpha^+$ of $\alpha^{++}$-algoritme, Fuzzy Miner, Genetic Miner, AGNEsMiner, DWS Mining, AWS Mining, FSM Miner/Petrify, Heuristic miner en Duplicates Genetic miner [5], [9], [10].

Vervolgens kunnen bedrijven de overeenkomst tussen de ontdekte processen en de vooropgestelde processen nagaan. Vooraleer men dit echter op een correcte en verantwoorde manier kan doen, is het belangrijk om te weten hoe correct de ontdekte modellen de event logs weergeven [10]. Dit kan gemeten worden met behulp van enkele dimensies die zijn verdeeld in twee omvattende domeinen, namelijk accuraatheid en begrijpbaarheid. Het eerste domein omvat drie dimensies, namelijk recall (sensitivity of fitness), precision en generalization. Het tweede domein bevat slechts twee dimensies, genaamd simplicity (complexity) en structuredness (understandability of entropy). Figuur 3 geeft hiervan een overzicht.




*Figuur 3: Overzicht van de kwaliteitsdimensies van een ontdekt procesmodel* [11]

Recall of fitness hoe goed het procesmodel in staat is om het geobserveerde gedrag op een correcte manier opnieuw af te spelen. Ten tweede is precision de capaciteit van het model om ongewenst gedrag te verwerpen. Vervolgens is generalization de kracht van het model om overfitting te vermijden. Simplicity geeft aan dat een eenvoudiger model verkozen moet worden boven een meer complex model, indien ze beide even goed in staat zijn om het geobserveerde gedrag weer te geven. Dit wordt ook Occam's Razor genoemd [12]. Structuredness, tot slot, geeft de gemakkelijkheid van interpretatie van het model weer [11].

Voor elk van deze dimensies zijn in de literatuur enkele metrieken ontwikkeld, die proberen de dimensies te kwantificeren. Ze proberen met andere woorden een interpreteerbare voorstelling te geven van de verschillende dimensies. De volgende opsomming geeft een overzicht van deze metrieken [4], [5], [8], [11]–[15]:

Fitness:

- Parsing measure
- Fitness
- Proper Completion
- Completeness
- Behavioural Recall
- Alignment Based Trace Fitness
- Behavioural Pofile metrics

Generalization:

- Generalization (L,M)
- Behavioural Generalization
- Weighted Behavioral Generalization
- Alignment Based Generalization (L,M)
- Alignment Based Probalistic Generalization
- Generalization

Precision:

- Soundness
- Simple behavioral appropriateness
- Advanced Behavioral Appropriateness
- Behavioural Specificity
- Behavioural Precision
- ETC Precision
- Precision(L,M)
- Alignment Based Precision
- One Align Precision
- Best Align Precision

Simplicity:

- Extended Cardoso Metric
- Extended Cyclomatic Metric
- Structuredness Metric
- Simple structural appropriateness
- Advanced structural appropriateness
- Het aantal:
  - Transities
  - Activiteiten
  - Bogen
  - Control-flow constructen

Omwille van het feit dat er zeer veel discovery technieken zijn, maar ook veel verschillende eigenschappen van procesmodellen, is het onduidelijk wat gemeten wordt door welke metriek en hoe elke metriek reageert op een bepaalde situatie. Het is niet geweten hoe gevoelig de metrieken zijn aan de keuze van het discovery algoritme of de complexiteit van het ontdekte model. Het is verder onduidelijk hoe de metrieken zich ten opzichte van elkaar verhouden en of ze hun respectievelijke dimensie wel degelijk kwantificeren.

## 1.2 Onderzoeksvragen

### 1.2.1 Centrale onderzoeksvraag

Zoals in de vorige sectie gesteld werd, zijn er veel metrieken om de kwaliteit van ontdekte procesmodellen te kwantificeren beschikbaar in de literatuur. In de literatuur wordt er een theoretische verklaring gegeven over hoe deze metrieken berekend worden. Wat echter onduidelijk is, is hoe deze metrieken zich ten opzichte van elkaar verhouden en wat ze nu juist meten. Er is niet geweten of de metrieken binnen een bepaalde dimensie wel degelijk deze dimensie meten. Verder kan ook de vraag gesteld worden wat mogelijke pijnpunten zijn van deze metrieken. Er zijn mogelijk situaties waar de metrieken zich anders gaan gedragen, of moeilijkheden hebben met het komen tot een oplossing. Dit leidt ons tot de volgende centrale onderzoeksvraag:

*Hoe verhouden de verschillende metrieken, gebruikt om de kwaliteit van procesmodellen te kwantificeren, zich tot elkaar en wat zijn hun mogelijke pijnpunten?*

### 1.2.2 Deelvragen

Vooraleer een antwoord geformuleerd kan worden op de centrale onderzoeksvraag, moeten eerst verschillende deelvragen beantwoord worden.

*Welke kwaliteitsdimensies worden voornamelijk gehanteerd in de literatuur en wat stellen ze voor?*

In de eerste plaats moet bepaald worden welke dimensies voornamelijk gehanteerd worden in de literatuur om de kwaliteit van de modellen voor te stellen. Om in een latere fase na te gaan of de onderzochte metrieken meten wat ze theoretisch beweren te meten, moet eerst onderzocht worden wat elk van de dimensies weergeeft.

*Welke metrieken bestaan er in de literatuur voor elk van de dimensies en wat meten ze?*

Vervolgens moet nagegaan worden welke metrieken er bestaan voor de verschillende kwaliteitsdimensies in de literatuur en hoe deze berekend worden. Verder moet onderzocht worden wat deze metrieken theoretisch zouden moeten meten en of ze wel degelijk hun respectievelijke dimensie voorstellen. Aan de hand van deze informatie kan de praktijk, die gesimuleerd wordt met behulp van artificiële data, vergeleken worden met de theorie.

*Hoe verhouden de metrieken zich tot elkaar?*

Gezien het grote aantal metrieken dat beschikbaar is, is de kans dat bepaalde metrieken sterk gerelateerd zijn aan elkaar reëel. Daarom wordt getracht een antwoord te formuleren op de vraag: gedragen de metrieken binnen één bepaalde dimensie zich gelijkaardig, of zijn er metrieken die eerder gerelateerd kunnen worden aan een andere dimensie? Deze analyse zal meer inzicht geven over wat de metrieken in werkelijkheid kwantificeren.

*Wat zijn mogelijke pijnpunten van de metrieken?*

Het is reeds geweten dat er enkele metrieken zijn die meer rekentijd nodig hebben dan andere, maar er kunnen eveneens andere situaties zijn waarin bepaalde metrieken moeilijkheden ondervinden met het komen tot een oplossing. Vandaar zal onderzocht worden welke mogelijke pijnpunten de metrieken kunnen bevatten.

## 1.3 Onderzoeksaanpak

In de eerste plaats moet een grondige literatuurstudie van wetenschappelijke artikels met betrekking tot process discovery en conformance checking gebeuren om een theoretische achtergrond op te bouwen. De referenties die gebruikt worden in deze literatuur gaan dieper in op aspecten die beantwoord moeten worden in de deelvragen.

Aan de hand van de literatuur worden de meest gebruikte kwaliteitsdimensies beschreven. Er zal worden weergegeven wat ze voorstellen en hoe ze zich theoretisch verhouden ten opzichte van elkaar. Dit zal een antwoord vormen op de eerste deelvraag.

Vervolgens zullen voor elk van deze dimensies de meest relevante metrieken, die gevonden werden in de literatuur, worden samengevat. Er zal onderzocht worden hoe deze metrieken berekend worden en wat ze volgens de literatuur kwantificeren. Door ze gestructureerd weer te geven kan een zo volledig mogelijk antwoord gevormd worden op de tweede deelvraag. Deze informatie kan eveneens gebruikt worden om een eerste zicht te krijgen op hoe de metrieken zich ten opzichte van elkaar verhouden, wat een gedeeltelijk antwoord biedt op de derde deelvraag.

Om de derde en de vierde deelvraag grondig te kunnen beantwoorden is een relatief grote hoeveelheid aan data vereist. Aangezien er in de praktijk niet veel reële event logs beschikbaar zijn, wordt de keuze gemaakt om gebruik te maken van artificiële event logs.

Aan de hand van de ontwikkelde event logs, gaan er procesmodellen ontdekt worden om hier vervolgens de gevonden metrieken op te berekenen. Om de metrieken te berekenen voor alle event logs kan gebruik gemaakt worden van ProM en het Cobefra framework. Beiden kunnen samenwerken met de VSC (Vlaams Supercomputer Centrum), waardoor de metrieken in batch kunnen berekend worden voor meerdere event logs.

Eens alle metrieken voor de verschillende event logs berekend zijn, kan de analyse beginnen. Mogelijke technieken hiervoor zouden bijvoorbeeld factor- en regressieanalyse kunnen zijn. De analyses moeten in staat zijn inzichten te verwerven in de gegenereerde data. Daarom moet er eerst gekeken worden of het mogelijk is deze technieken toe te passen in deze onderzoekssetting. Literatuur uit de multivariate data-analyse kan hier meer duidelijk scheppen.

Als de methodes gekozen zijn, kan er gebruik gemaakt worden van statistische software zoals SPSS of R om de analyses uit te voeren. Deze software kan de nodige ondersteuning bieden tijdens de analyse om zo inzichten te verwerven in de data en een antwoord te vormen op de deelvragen.

## 1.4 Bedrijfsrelevantie

Deze masterproef is bedrijfseconomisch gezien relevant omdat het voor bedrijven belangrijk is om hun processen zo goed mogelijk te optimaliseren. Vooraleer dit echter bereikt kan worden, moeten er uit hun event logs procesmodellen gegenereerd worden. Vervolgens is het mogelijk om op deze procesmodellen eventuele verbeteringen toe te passen. Nochtans moeten de bedrijven eerst zekerheid hebben dat die modellen conform hun event-logs zijn alvorens ze hun aanpassingen maken. Dat is waar de kwaliteitsdimensies, gemeten door de metrieken, in het spel komen. Ze worden gebruikt om de mate van overeenkomst tussen de event log en het procesmodel te meten.

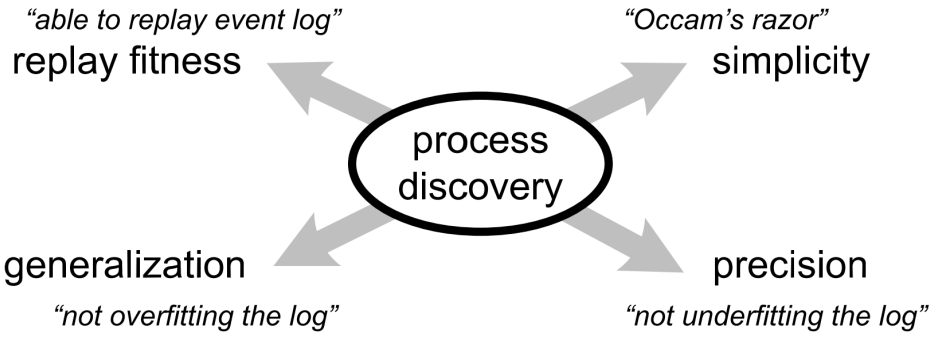
De analyse van deze metrieken, gemaakt in deze masterproef, zal de bedrijven meer zekerheid geven over welke metrieken ze in welke situatie kunnen gebruiken. Verder zal meer duidelijkheid geschept worden over hoe de metrieken zich ten opzichte van elkaar verhouden. Zo kunnen de bedrijven, indien gebruik gemaakt wordt van de geanalyseerde metrieken, over meer zekerheid beschikken wanneer er beslissingen genomen worden omtrent eventuele procesverbeteringen. Als de bedrijven metrieken zouden gebruiken die niet geschikt zijn voor bepaalde situaties, zou dit kunnen leiden tot foute beslissingen met betrekking tot hun processen. Dit zou op termijn kunnen leiden tot significante financiële verliezen. Tot slot zal het makkelijker zijn om conclusies te trekken in verband met de kwaliteit van hun procesmodellen, waardoor ze met meer betrouwbaarheid hun activiteiten kunnen uitvoeren.

# Hoofdstuk 2: De evaluatiemetrieken

In dit hoofdstuk worden de verschillende dimensies besproken die in de literatuur gehanteerd worden om de kwaliteit van een procesmodel te beoordelen. Vervolgens worden voor elke van deze dimensies enkele van de meest relevante model-log metrieken in detail besproken. Er wordt dus niet gekeken naar model-model metrieken die de kwaliteit van een ontdekt procesmodel gaan vergelijken met het vooropgestelde model. Verder zal eveneens niet gekeken worden naar de metrieken gebruikt voor de kwaliteitsdimensie simplicity, omdat er bij deze dimensie enkel gekeken wordt naar het ontdekte model. In deze dimensie wordt dus geen rekening gehouden met hoe dit model zich verhoudt tot de event log.

## 2.1 Kwaliteitsdimensies

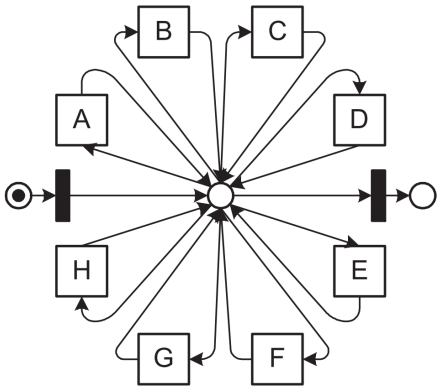
Om de algemene kwaliteit van een procesmodel te beschrijven, worden er in de literatuur voornamelijk vier dimensies gehanteerd [5], [11], [13], [14], [16]–[18]. Elk van deze dimensies dekt een ander aspect van de kwaliteit van een procesmodel. Merk op dat in het vorige hoofdstuk gesproken werd over vijf dimensies. Daar werd structuredness als een aparte dimensie beschouwd, maar deze dimensie wordt in de literatuur vaak samengenomen met simplicity. Onderstaande figuur geeft een overzicht van de vier gehanteerde dimensies [14].




*Figuur 4: Kwaliteitsdimensies van process discovery* [14]

De dimensie fitness, ook wel recall genoemd, beschrijft het gedrag in de event log dat opnieuw kan worden afgespeeld door het procesmodel [14], [19]. Een model met een perfecte fitness is in staat om alle traces in de log af te spelen op het model van begin tot einde [20]. Het gaat met andere woorden na hoe goed het gedrag in de event log, werd opgenomen in het procesmodel [15].

Precision daarentegen, schat het gedrag dat toegelaten wordt door het procesmodel dat niet geobserveerd wordt in de event log. Een model is precies, als het niet 'te veel' gedrag toelaat. Een duidelijk voorbeeld van een model dat een gebrek heeft aan precision is het *flower model* (Figuur 5), omdat in dit model oneindig veel gedrag mogelijk is, waardoor dit nooit volledig kan worden weergegeven in een event log [13], [20]. In het flower model kan namelijk na het uitvoeren van een activiteit, eender welke activiteit uitgevoerd worden. Omdat er in een procesmodel dus mogelijk ongelimiteerd gedrag voorkomt, zoals in een flower model of in het geval van loop, kan er slechts een schatting van precision gemaakt worden [14]. Precision gaat er dus voor zorgen dat een bepaald model niet underfitting is voor een gegeven event log. Underfitting is het probleem dat een ontdekt model te veel gedrag toelaat dat niet werd geobserveerd in de event log of met andere woorden dat het model het gedrag in de log gaat over veralgemenen [21].



*Figuur 5: Flower model*

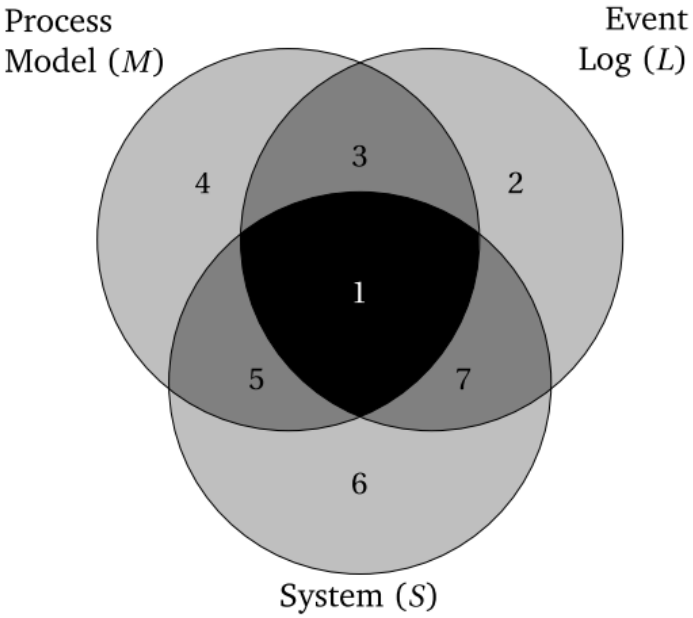
Generalization geeft aan of het model al dan niet overfitting is voor het gedrag dat gevonden wordt in de event log [14]. Overfitting is het probleem dat een zeer specifiek model wordt ontdekt, dat geen gedrag toelaat dat niet wordt geobserveerd in de event log [4]. Dit is het tegenovergestelde van underfitting. Generalization gaat dus weergeven hoeveel gedrag het model toelaat dat niet werd geobserveerd in de event log. Het model moet namelijk voldoende veralgemenen, zodat het zich niet enkel beperkt tot de voorbeelden die in de log gevonden worden [20]. Het beoordeelt, met andere woorden, de mate waarin het procesmodel in staat is om toekomstig gedrag van het proces te reproduceren [13], [20].

Simplicity, ten slotte, evalueert hoe eenvoudig een procesmodel te begrijpen is [14], of met andere woorden het kwantificeert de complexiteit van het model [13]. Het meest eenvoudige model dat het gedrag verklaard dat gezien wordt in de event log, is het beste. Dit principe is eveneens gekend als Occam's Razor. [20]

Het is belangrijk om op te merken dat het vaak een afweging is tussen deze dimensies. Zo is het bijvoorbeeld niet mogelijk om een procesmodel te ontdekken dat zeer hoog scoort op de dimensie precision en tegelijkertijd hoog scoort op de dimensie generalization. Hetzelfde geldt voor fitness en simplicity, modellen die zeer hoog scoren op fitness zijn in het geval van een realistisch event log doorgaans zeer complex en scoren dus laag op simplicity. Er moet dus gezocht worden naar een model dat de verschillende dimensies het beste afweegt ten opzichte van elkaar. [2]

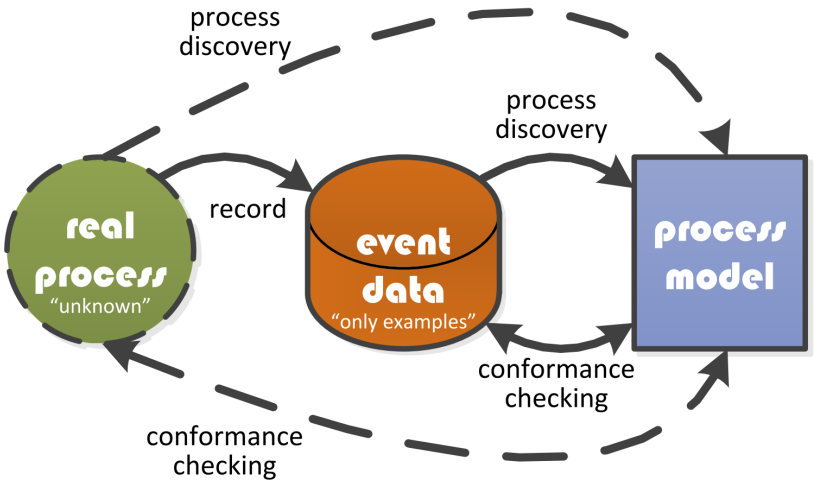
In de literatuur worden vele metrieken voorgesteld om de kwaliteit van de ontdekte procesmodellen te kwantificeren. Voor elk van de vier kwaliteitsdimensies zijn een reeks metrieken ontwikkeld die op verschillende manieren proberen de kwaliteitsdimensies van het model te kwantificeren. Nochtans is het op dit moment onduidelijk wat veel van deze metrieken precies proberen te kwantificeren en/of hoe ze dit doen. Voor deze masterproef wordt gefocust op enkele metrieken voor de dimensies

fitness, precision en generalization. Deze drie dimensies gebruiken het gedrag dat werd opgenomen in de event log om het procesmodel te evalueren. [22]




*Figuur 6: Venn diagram met het procesmodel (M), event log (L) en systeem (S)* [22]

In bovenstaand Venn diagram wordt gebruik gemaakt van het procesmodel, de eventlog en het systeem om zeven gebieden te kunnen onderscheiden. Het systeem is het gedrag dat voorkomt in het onderliggende proces. In de realiteit is er altijd gedrag dat voorkomt in het systeem, maar dat niet bekend is. De reden hiervoor is dat er geen manier is om het gedrag van een systeem expliciet te beschrijven. Het gedrag van een systeem is typisch oneindig en in een real-world systeem is het altijd mogelijk dat onvoorzien gedrag voorkomt dat eventueel kan veranderen doorheen de tijd [14]. Een event log is dus slechts een mogelijke voorstelling van het gedrag in het systeem, het bevat met andere woorden slechts voorbeeldgedrag van het systeem. Het model verklaart vervolgens de betreffende voorbeeldlog, maar een volgende voorbeeldlog van hetzelfde systeem kan een totaal verschillend procesmodel produceren. Het systeem wordt daarom gedefinieerd als de verzameling van al het mogelijke gedrag dat voorgesteld kan worden door verschillende event logs [2]. Onderstaande figuur geeft een duidelijk overzicht hierover. Het systeem, of het werkelijke proces, wordt voorgesteld door event logs. Deze event logs zijn slechts mogelijke voorstellingen van het systeem. Vervolgens kunnen deze event logs gebruikt worden om procesmodellen uit te ontdekken, waarmee uiteindelijke conformance checking kan uitgevoerd worden.




*Figuur 7: Het werkelijke proces ten opzichte van de event log en het procesmodel* [20]

De bedoeling van process discovery is om een procesmodel te vinden, dat het systeem zo accuraat mogelijk beschrijft door gebruik te maken van het geobserveerde gedrag in de event log. Aan de hand van de event log kunnen dan metrieken opgesteld worden die de verschillende kwaliteitsdimensies tussen het model en de log proberen uit te drukken. [22]

Zoals eerder vermeld, kunnen in het Venn diagram zeven gebieden onderscheiden worden. Bijvoorbeeld gebied 1 is het gebied dat al het gedrag bevat van het systeem dat ook wordt geobserveerd in de event log en mogelijk is volgens het procesmodel. Gebied 3 zijn de uitzonderingen die geobserveerd worden in de event log en beschreven worden door het procesmodel, maar die niet voorkomen in het systeem [22]. Deze gebieden worden gebruikt om de verschillende kwaliteitsdimensies te identificeren.

De fitness tussen het model en de event log geeft het gedrag weer van de event log dat getoond wordt in het procesmodel, vergeleken met al het geobserveerde gedrag in de event log [22]:

$$Model - \log fitness = \frac{|L \cap M|}{|L|}$$

Hoe meer gedrag van de event log dat ook toegelaten is door het model, hoe beter de fitness. Voor deze dimensie worden de Token Based Fitness, Behavioral Recall en de Alignment Based Trace Fitness besproken (sectie 2.2).

De precision tussen het procesmodel en de event log drukt de hoeveelheid gedrag uit dat kan geproduceerd worden door het procesmodel, dat in de event log wordt geobserveerd [22]:

$$Model - \log precision = \frac{|L \cap M|}{|M|}$$

Hoe meer gedrag door het model wordt weergegeven dat niet in de event log staat, hoe lager de precision is. Voor deze metriek worden de Behavioral Precision, ETC Precision, One Align Precision, Best Align Precision en de Alignment Based Precision besproken (sectie 2.3).

Generalization probeert in te schatten hoe goed het procesmodel het gedrag van het (onbekende of niet geobserveerde) systeem beschrijft en dus niet enkel de event log met het gedrag van het geobserveerde systeem [22]:

$$Model - system\ recall = \frac{|S \cap M|}{|S|}$$

Hoe meer gedrag van het systeem wordt weergegeven door het model, hoe beter de generalization van het model is. Voor deze laatste dimensie worden ook drie metrieken besproken, namelijk de Weighted Behavioral Generalization en de Alignment Based Generalization (sectie 2.4).

Zoals blijkt uit bovenstaande definities, wordt duidelijk dat fitness geen invloed zou mogen hebben op precision. De hoeveelheid traces in de log die niet voorkomen in het model (gebied 2 en 7), zou geen invloed mogen hebben op de berekening van de model-log precision. Zolang de doorsnede tussen de log en het model (gebied 1 en 3) gelijk blijft, zou precision niet mogen veranderen.

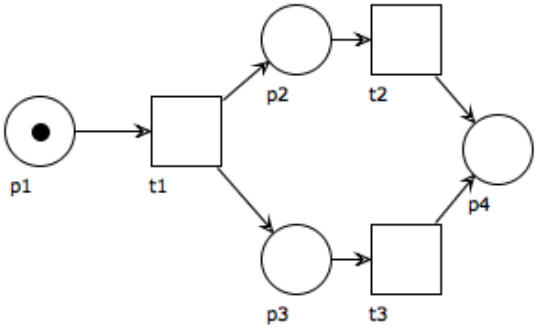
De verhouding tussen generalization en precision kan ook verklaard worden aan de hand van bovenstaande definities. Als bijvoorbeeld een zeer precies model gewenst wordt, zullen gebied 1 en 3 zo groot mogelijk moeten zijn. Dit gaat ten koste van de grootte van gebied 5, aangezien er bij precision geen rekening gehouden wordt met het systeem. Er wordt dan een model gezocht dat zo min mogelijk gedrag weergeeft dat niet voorkomt in de log (gebieden 4 en 5). Hierdoor zal generalization echter dalen, omdat gebied 5 (samen met gebied 1) de hoogte van generalization bepaald. Het omgekeerde geldt uiteraard ook.

Om de metrieken die besproken zullen worden te kunnen berekenen, moet het model voorgesteld worden door een Petri net. Petri nets laten namelijk toe om de staat van een proces weer te geven doormiddel van tokens, wat nodig is om de onderstaande metrieken te berekenen. Voor het berekenen van enkele van de besproken metrieken wordt onder andere gekeken naar het aantal geproduceerde, geconsumeerde, achterblijvende of ontbrekende tokens. Petri nets zijn in staat om elk van deze voor te stellen [2]. Er is ook een praktische reden verbonden aan het gebruik van Petri nets. Zoals in hoofdstuk 3 zal besproken worden, zal gebruik gemaakt worden van ProM om de algoritmes toe te passen op de event logs. De output van deze algoritmes zal in de vorm van Petri nets zijn.

Petri nets zijn een mogelijke manier om procesmodellen voor te stellen op een relatief simpele en intuïtieve manier. Een Petri net wordt opgebouwd uit plaatsen (cirkels) en transities (vierkanten). De structuur is in essentie statisch, maar met behulp van de zogeheten firing rules, kunnen tokens doorheen het modellen bewegen. De staat van een Petri net wordt bijgevolg weergegeven door de verdeling van deze tokens over de plaatsen. De firing rule wordt als volgt gedefinieerd: een transitie is toegelaten (*enabled*) om uitgevoerd te worden (fire) als elk van zijn input-plaatsen een token bevatten. Als dit gebeurt worden de tokens in elk van zijn input-plaatsen geconsumeerd en tokens geproduceerd in elk van zijn output-plaatsen. [2]

Om dit te illustreren aan de hand van een voorbeeld kan figuur 8 beschouwd worden. In dit voorbeeld is transitie t1 enabled, aangezien er zich in plaats p1, de enige input-plaats van t1, een token bevindt.

Als t1 wordt uitgevoerd, zal de token in p1 geconsumeerd worden en vervolgens tokens produceren in zowel p2 als p3.




*Figuur 8: Voorbeeld Petri net*

## 2.2 Metrieken gebruikt voor fitness

### 2.2.1 Token Based Fitness

Token Based Fitness wordt bekomen door te evalueren of elke trace in de event log kan gereproduceerd worden door het model. Deze procedure wordt ook wel trace replay genoemd. Tijdens deze replay zal elke transitie tokens consumeren alvorens de activiteit uit te voeren en nadien tokens produceren. Elke token die extra geproduceerd moet worden en elke token die na het herhalen te veel overblijft wordt afgestraft in deze metriek. [11], [20]

Een event log en een Petri net behalen een goede score op fitness als het Petri net elk trace in de event log kan herhalen zonder dat extra tokens gegenereerd moeten worden of zonder tokens die na het vervolledigen van de trace achter blijven in het model. Als dit het geval is, wordt een fitness van 1 bekomen. [8]

De formule om fitness te berekenen wordt voorgesteld in [8], [20]:

$$f = \frac{1}{2}\left(1 - \frac{\sum_{i=1}^{k} n_i m_i}{\sum_{i=1}^{k} n_i c_i}\right) + \frac{1}{2}\left(1 - \frac{\sum_{i=1}^{k} n_i r_i}{\sum_{i=1}^{k} n_i p_i}\right)$$

Waarbij:

$k$ = aantal verschillende traces van de geaggregeerde log

$n_i$ = frequentie van trace i

$m_i$ = aantal ontbrekende tokens voor elk log trace i

$r_i$ = aantal tokens die overblijven voor elk log trace i

$c_i$ = aantal geconsumeerde tokens voor elk log trace i

$p_i$ = aantal geproduceerde tokens tijdens de log replay van de huidige trace i

Merk op dat voor elke i, $m_i \leq c_i$ en $r_i \leq p_i$, waardoor $0 \leq f \leq 1$. Verder zijn $c_i$ en $p_i$ altijd groter dan 0, omdat altijd minstens één token geproduceerd zal worden voor de start plaats en één token

geconsumeerd bij de eind plaats. Hoe hoger deze waarde, hoe beter de dimensie fitness is volgens deze metriek. [2]

Deze metriek kan echter problemen ondervinden indien bijvoorbeeld meerdere transities voorkomen met hetzelfde label of transities die onzichtbaar zijn. Het algoritme weet in die gevallen niet welk pad gevolgd moet worden indien er twee transities met hetzelfde label enabled zijn. [17]

### 2.2.2 Behavioral Recall

Als een event log beschikbaar is, die aangevuld wordt met artificiële negatieve events, kan een confusion matrix (Tabel 1) opgesteld worden. Negatieve events registreren dat op een bepaalde positie in een trace, een bepaald event niet kan voorkomen [23]. De matrix wordt opgesteld door elke trace in de event log opnieuw af te spelen op het model. Als er een positief event wordt tegengekomen, wordt er gecontroleerd of een activiteit kan gevonden worden in het model om het corresponderende positieve event uit te voeren. Voor Petri nets, wordt dus nagegaan of een transitie, die toegewezen wordt aan het positieve event, enabled is. Als dit het geval is wordt True Positive (TP) verhoogd met 1. Als dit niet het geval is, wordt dit event toegevoegd aan de set van False Negatives (FN), gevolgd door het geforceerd afvuren van deze disabled transitie, die toegewezen is aan het positieve event. Negatieve events worden anders behandeld dan de positieve bij het evalueren van een procesmodel. Er wordt gecontroleerd of het mogelijk is om een transitie, die toegewezen is aan een negatief event, af te vuren. Als dit het geval is, wordt False Positive (FP) verhoogd met 1, indien niet wordt True Negative (TN) verhoogd met 1 [12]. De opgestelde confusion matrix kan vervolgens gebruikt worden om enkele metrieken te berekenen, zoals de Behavioral Recall en de Behavioral Precision (sectie 2.2.1).

| | Actual Positive | Actual Negative |
|---|---|---|
| Predicted Positive | True Positive (TP) | False Positive (FP) |
| Predicted Negative | False Negative (FN) | True Negative (TN) |

*Tabel 1: Confusion matrix*

Behavioral Recall wordt gedefinieerd als het percentage van correct geclassificeerde positieve events in de event log (TP) ten opzichte van alle werkelijk positieve events (TP + FN). Ook bij deze metriek wordt dus de trace replay techniek gebruikt. Hiermee wordt geverifieerd of elk positief event geparsed kan worden door het model. Deze metriek is verschillend van fitness omdat het zich focust op de events die succesvol afgespeeld kunnen worden door het procesmodel, zonder dat het rekening houdt met het exacte aantal ontbrekende of achterblijvende tokens in het model. [23], [24]

TP en FN worden geïnitialiseerd op 0 om vervolgens alle traces achtereenvolgens te parsen. Wanneer een toegestane transitie uitgevoerd wordt, wordt de waarde van TP verhoogd met 1. Deze transities komen overeen met de transities in gebied 1 en 3 van het Venn diagram. Als een transitie die niet toegestaan is, geforceerd moet worden om afgevuurd te worden, wordt FN verhoogd met 1. Deze

transities komen overeen met gebied 2 en 7 van het Venn diagram. Aan de hand van deze twee waardes kan de formule voor Behavioral Recall als volgt gedefinieerd worden [23]:

$$r_B^p = \frac{\sum_{i=1}^{k} n_i TP_i}{\sum_{i=1}^{k} n_i TP_i + \sum_{i=1}^{k} n_i FN_i}$$

Waarbij:

$k$ = aantal traces

$\mathrm{n_i}$ = aantal process instanties

$\mathrm{TP_i}$ = aantal events die correct geparsed worden voor trace i

$\mathrm{FN_i}$ = aantal events waarvoor een transitie geforceerd werd om af te vuren voor trace i

Behavioral Recall evolueert tussen 0 en 1. Als er enkel TP's gevonden worden en geen FN's zal $r_B^p$ 1 zijn. Dit wil zeggen dat elke activiteit die afgevuurd werd, ook daadwerkelijk toegestaan was. Als geen enkel event toegestaan was, en dus elke activiteit geforceerd werd om afgevuurd te worden, zal deze metriek 0 bedragen. Hoe dichter bij 1, hoe meer gedrag in de event log overeenkomt met die in het model en dus hoe beter, volgens deze maatstaf, de kwaliteitsdimensie fitness is. Met andere woorden, hoe dichter de $r_B^p$ bij 1 ligt, hoe meer activiteiten correct konden worden afgespeeld op het model. Een waarde van 0,6 voor deze metriek zou bijgevolg kunnen geïnterpreteerd worden als 60% van alle uitgevoerde events, werden op een correcte manier uitgevoerd.

### 2.2.3 Alignment Based Fitness

De volgende metriek baseert zich op het aligneren van het procesmodel met de event log, in plaats van de log opnieuw af te spelen in het model zoals bij bijvoorbeeld de Behavioral Recall. Elke trace kan gealigneerd worden met verschillende paden in het model. Er wordt echter voor elke trace gezocht naar de meest optimale alignment met het model, dit is de alignment die de minste bewegingen nodig heeft om de trace af te spelen op het model. Een alignment is een opeenvolging van bewegingen die toegelaten worden, om ervoor te zorgen dat elke activiteit in een trace overeenkomt met een activiteit in het model [20]. De metriek straft af voor alignments die een beweging nodig hebben in het model, zonder een corresponderende beweging in de log en omgekeerd [11]. Aan elke beweging die vereist is tijdens het aligneren, wordt een bepaalde kost[1] gekoppeld. Om deze kost te berekenen kan gebruik gemaakt worden van bijvoorbeeld onderstaande tabel:

| **Trace** | a | b | d | c | >> | e | f | e | g |
|---|---|---|---|---|---|---|---|---|---|
| **Model** | a | b | >> | c | d | e | >> | >> | g |

*Tabel 2: Alignment voorbeeld*

[1] Deze kosten kunnen afhangen van het type activiteit. Zo is bijvoorbeeld het overslaan van een betalingsactiviteit ernstiger dan het versturen van te veel brieven. [46]

De bovenste rij stelt alle activiteiten voor in een trace uit een event log. De onderste rij stelt een trace voor die voorkomt in het model. In dit voorbeeld kan d niet correct herhaald worden omdat het te vroeg voorkomt in de trace, terwijl er geen activiteit d voorkomt op de derde plaats in het model. Dit wordt weergegeven door het >>-symbool. Als deze situatie voorkomt wordt het een *move on log only* genoemd. Later in de trace moet activiteit d volgens het model wel uitgevoerd worden, maar gebeurt dit niet in de log. Deze situatie wordt een *move on model only* genoemd. In het voorbeeld komen er dus in totaal één *move on model only* en drie *moves on log only* voor. Elk voorkomen kan afgestraft worden met een bepaalde kost die gebruikt wordt om de volgende formule te berekenen [22].

Een mogelijke formule voor het berekenen van deze metriek wordt voorgesteld in [22]:

$$Q_{\mathrm{rf}} = 1 - \frac{\sum_{traces} cost\ for\ aligning\ model\ and\ trace \times trace\ frequency}{Cost\ to\ align\ log\ on\ model\ with\ no\ synchronous\ moves}$$

De noemer in de formule is de maximale kost die kan voorkomen voor een bepaalde combinatie van een trace en een model. Dit wordt gebruikt om de Alignment Based Fitness te normaliseren tot een waarde die ligt tussen 0 en 1. Hoe korter deze waarde bij 0 ligt, hoe hoger de kost is voor het aligneren van het model met de log en dus hoe lager de fitness is. Omgekeerd, hoe korter bij 1, hoe beter de fitness zal zijn, omdat er dan geen kost is voor het aligneren van het model met de log, wat betekent dat elke trace correct afgespeeld kan worden op het model zonder dat hier bewegingen voor nodig zijn. [22]

De methode van Alignment Based Fitness tracht een oplossing te bieden aan enkele van de tekortkomingen van de Token Based Fitness. Zo zijn er bij het gebruik van de Alignment Based Fitness geen problemen meer als er meerdere transities voorkomen met hetzelfde label of transities die onzichtbaar zijn [17]. Verder wordt er in deze metriek de nadruk gelegd op de taken die enabled worden en niet op het aantal tokens dat geproduceerd of geconsumeerd worden. Hierdoor heeft het bij deze metriek geen invloed of taken één of drie preceding tasks nodig hebben om de volgende activiteit te enablen. Met andere woorden, het model heeft hier minder invloed op de berekening van de metriek. Nochtans, in het geval van een lage fitness zal de Petri net tijdens de replay *overspoeld* worden met tokens, omdat in dat geval het model en de event log minder goed overeenstemmen. Aangezien het aantal tokens geen invloed heeft op de berekening zal deze metriek een meer optimistische schatting geven van de fitness.

## 2.3 Metrieken gebruikt voor precision

### 2.3.1 Behavioral Precision en Weighted Behavioral Precision

Ook deze metriek maakt, zoals Behavioral Recall (sectie 2.2.2), gebruik van de confusion matrix (tabel 1). Er wordt in dit geval een verhouding berekend tussen de TP's en de FP's. Er wordt met andere woorden enkel gekeken naar die activiteiten die positief worden voorspeld en dus toegelaten zijn in het model. Dit zijn de activiteiten die zich bevinden in gebied 1 en 3 van het Venn diagram.

De FP's zijn die activiteiten die zich bevinden in gebied 4 en 5 van het Venn diagram, met andere woorden activiteiten die voorkomen in het model, maar niet in de event log. Op deze manier kan een idee gevormd worden van de precision-dimensie van een procesmodel. Onderstaande formule berekent aan de hand hiervan de Behavioral Precision, zoals voorgesteld in [25]:

$$p_b = \left( \frac{\sum_{i=1}^{k} n_i TP_i}{\sum_{i=1}^{k} n_i TP_i + \sum_{i=1}^{k} n_i FP_i} \right)$$

Waarbij:

$k$ = het totale aantal verschillende gegroepeerde proces instanties in de event log

$i$ = loopt over alle verschillende gegroepeerde proces instanties

$n_i$ = aantal instanties binnen een groep van gelijkaardige proces instanties

$TP$ = aantal correct voorspelde positieve events

$FP$ = aantal fout voorspelde artificiële negatieve events

Net als bij Behavioral Recall, zal ook deze metriek tussen 0 en 1 liggen. Als alle events voorkomen in zowel het model als in de log en er dus enkel TP's worden waargenomen, zal deze metriek 1 bedragen. Omgekeerd, als alle artificiële negatieve events worden voorspeld als positieve events (FP) en er dus geen activiteiten worden geobserveerd zowel in het model als in de log, zal deze metriek 0 bedragen. Hoe hoger naar 1, hoe beter de dimensie precisie voor een bepaald model is. Stel dat deze metriek 0,88 bedraagt, dan wil dit zeggen dat 88% van de events die voorspeld worden positief te zijn, dit ook daadwerkelijk zijn [2]. Met andere woorden, 88% van de events die voorkomen in het model worden ook waargenomen in de log.

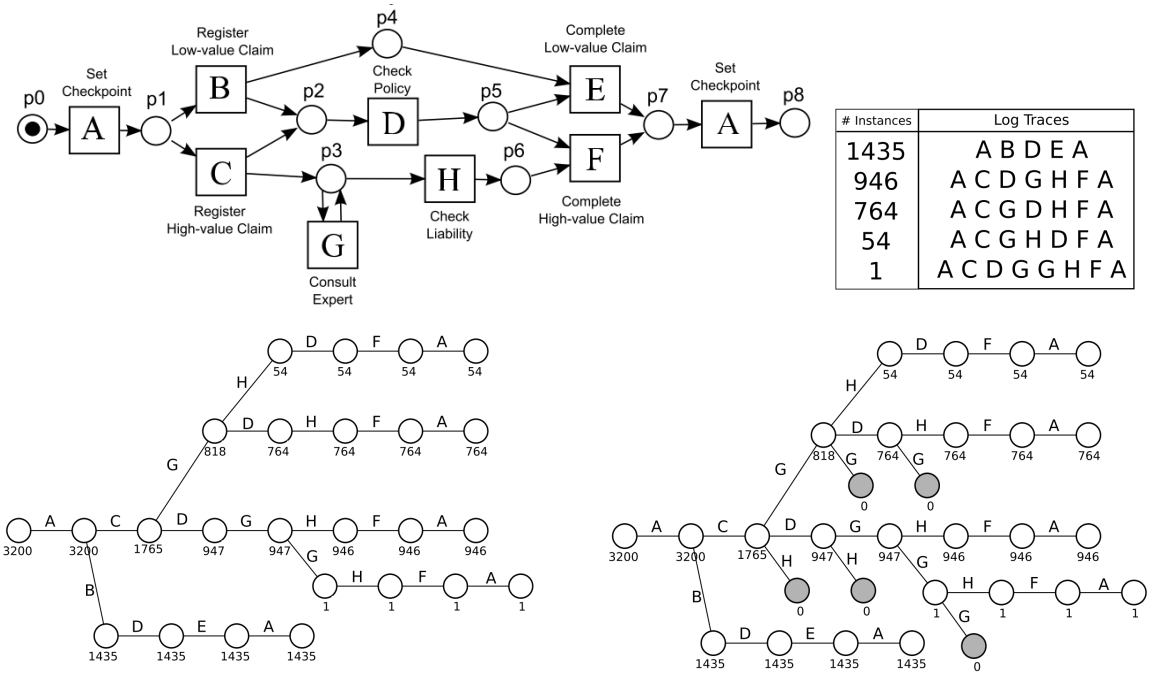
Er bestaat eveneens een gewogen variant van deze metriek waarbij elk negatief event gewogen wordt. Hoe langer een overeenkomend prefix kan gevonden worden die gelijk is aan de prefix van het negatieve event dat beschouwd wordt, hoe lager het gewicht dat aan het negatieve event gegeven wordt. Bij een gewicht van 0, wat het minimum is, betekent dit dat een trace werd gevonden die dezelfde prefix heeft als gezien bij het negatieve event, wat wil zeggen dat dit gedrag werkelijk is voorgekomen en dus niet kan worden aanschouwd als zijnde niet toegelaten of negatief gedrag. Bij een gewicht van 1, wat het maximum is, is dit een indicatie dat het gedrag voorgesteld door het negatieve event inderdaad niet mag toegelaten worden. [12]

### 2.3.2 Alignment Based Precision

Voor de Alignment-Based Precision wordt er in [4] gekeken naar elk event $e \in \mathcal{E}$ ($\mathcal{E}$ is de verzameling van unieke events $e$) dat impliciet verwijst naar een punt in de event log juist voor het uitvoeren van event $e$. Er wordt een verhouding gemaakt tussen het aantal activiteiten die enabled zijn in het model $en_M(e)$ en het aantal geobserveerde activiteiten die daadwerkelijk zijn uitgevoerd in eenzelfde context $en_L(e)$. Onder de context van een event e, wordt de activiteitenprefix van de process instantie verstaan die juist voor event e voorkomt. Met andere woorden, de trace van activiteiten die voorkomen net voor het uitvoeren van event e. Deze context is vergelijkbaar met het

automaton dat verder besproken wordt bij respectievelijk One Align (sectie 2.3.4) en Best Align Precision (sectie 2.3.5).

Een automaton wordt opgesteld door alle traces in de event log te doorlopen en deze traces te tekenen zoals weergegeven in Figuur 9. Op deze manier kan het gedrag dat voorkomt in de event log op een duidelijke manier weergegeven worden. Vervolgens wordt deze prefix automaton uitgebreid met het gedrag dat voorkomt in het procesmodel (Figuur 9). Op deze manier kan de extended prefix automaton geconstrueerd worden. De grijze bolletjes in Figuur 9 tonen het extra gedrag dat voorkomt in het model. De extended prefix automaton toont dus zowel het gedrag dat voorkomt in de event log, als het gedrag dat voorkomt in het procesmodel. [26]




*Figuur 9: Een procesmodel en event log waaruit de prefix en extended prefix automaton geleerd worden*

Door de prefix automaton te vergelijken met de extended prefix automaton, kunnen situaties gevonden worden waarin het model afwijkt van het gedrag in de log. Deze afwijkingen worden escaping edges genoemd. Escaping edges zijn die situaties waarin het model meer gedrag toelaat dan de event log (grijze bolletjes), en waar het model dus minder precies is. [26]

De formule die vervolgens voorgesteld wordt in [4] is de volgende:

$$precision(L, M) = \frac{1}{|\varepsilon|} \sum_{e \in \mathcal{E}} \frac{|en_L(e)|}{|en_M(e)|}$$

Als al het gedrag dat toegelaten wordt door het model ook daadwerkelijk geobserveerd wordt in de event log, dan zal deze precision-metriek 1 bedragen. Aangezien er een gemiddelde berekend wordt over alle events heen, wordt automatisch rekening gehouden met frequenties. Het is namelijk zo, dat als een activiteit die enabled is op een frequent pad, maar die nooit uitgevoerd wordt, harder

gestraft wordt dan een ongebruikte activiteit die toegelaten is op een infrequent pad. Als er enkel gedrag wordt toegelaten door het model, maar er van dit gedrag niets geobserveerd wordt in de event log, dan zal deze metriek 0 bedragen. [4]

Deze metriek is gelijkaardig aan de One Align en Best Align Precision, met het verschil dat er voor de Alignment Based Precision geen afzonderlijk gewicht wordt toegekend per trace, maar er een gemiddelde over de hele log berekend wordt op basis van de frequentie van elke trace.

### 2.3.3 ETC Precision

In [18] wordt er gebruik gemaakt van de escaping edges (zoals uitgelegd in sectie 2.3.2), als indicator voor het meten van het gedrag van een model in vergelijking tot het gereflecteerde gedrag in de event log. ETC Precision gaat de hoeveelheid escaping edges ($E_E$) vergelijken met het totale aantal toegelaten taken ($A_T$) in het model, om zo een idee te krijgen van de precision van het model. Om de ETC Precision te kunnen berekenen, wordt de assumptie gemaakt dat de event log en het model een perfecte fitness hebben (fitness = 1). In [18] wordt volgende formule voorgesteld:

$$etc_P(EL, PN) = 1 - \frac{\sum_{i=1}^{|\mathrm{EL}|} \sum_{j=1}^{|\sigma_i|+1} |E_E(s_j^i)|}{\sum_{i=1}^{|\mathrm{EL}|} \sum_{j=1}^{|\sigma_i|+1} |A_T(s_j^i)|}$$

Waarbij:

EL = $\{\sigma_1, \dots, \sigma_{|\mathrm{EL}|}\}$, een event log

PN = (P,T,W,$M_0$), een Petri net

$\sigma_i$ $(1 \leq i \leq |\mathrm{EL}|)$, voor elk trace i

$s_j^i$ $(1 \leq j \leq |\sigma_i| + 1)$, de staat van trace i

In deze formule wordt de set van escaping edges gedeeld door de set van de toegelaten taken in het model. Zo evalueert de metriek de hoeveelheid van overschatting in elk trace. Merk op dat voor alle $|E_E(s_j^i)| \leq |A_T(s_j^i)|$, waardoor $0 \leq etc_p \leq 1$. Hoe minder escaping edges er in het model zitten, hoe korter de metriek naar 1 zal liggen en bijgevolg hoe beter de precision van het model is. Er wordt in deze metriek rekening gehouden met de frequentie van de traces, waardoor de meest gebruikte en gepaste traces zwaarder doorwegen dan de meer uitzonderlijke traces die minder vaak voorkomen. [18]

De twee volgende metrieken die worden besproken (One Align Precision en Best Align Precision) breiden de ETC Precision uit. Deze metrieken gaan eerst het model met de event log aligneren om zo het grootste probleem van de ETC Precision op te lossen, namelijk de assumptie dat een log een fitness van 1 heeft voor een bepaald model. One Align Precision (sectie 2.3.4) en Best Align Precision (sectie 2.3.5) gaan door het aligneren van de event log met het model deze assumptie afdwingen. [11]

### 2.3.4 One Align Precision

Zoals vermeld in sectie 2.3.2 moet, voor zowel deze als de volgende metriek, eerst het model gealigneerd worden met de event log. In [27] wordt hiervoor een methode voorgesteld. Voor elke combinatie van een log met een model zijn er uiteraard meerdere mogelijkheden om deze te aligneren. Daarom berekent men een kost per alignment op basis van de afstand tussen het model en de event log. Hoe vaker er een move on model/log voorkomt tijdens de alignering, hoe groter de afstand tussen het model en de event log. Vervolgens wordt een zogeheten afstandsfunctie gebruikt om de kost te bepalen. Deze functie kan gedefinieerd worden door de gebruiker, maar standaard wordt er een afstandsfunctie gehanteerd die een eenheidskost toekent aan elke move on log/model only.

Met deze methode kan dus één optimale alignment ($\mathcal{A}^1$) of alle optimale alignments ($\mathcal{A}$) gevonden worden voor elk trace in de log. Met behulp van een enkele optimale alignment kan er reeds een waarde berekend worden voor precision. Nochtans ligt de waarde gebaseerd op alle optimale alignments korter bij de realiteit. Het probleem is dat deze laatste voor grote event logs moeilijker te berekenen is, omwille van een veel grotere vereiste rekencapaciteit. Daarom wordt vaak de One Align Precision gebruikt als benadering voor de waarde die bekomen zal worden als alle optimale alignments gebruik worden.

De formule die voorgesteld wordt in [27] om de One Align Precision te berekenen is in feite dezelfde als die van ETC Precision, maar er wordt een andere notatie gebruikt. Ook het gewicht dat in deze formule wordt gebruikt, zit verwerkt in de formule van ETC Precision, omdat beide zijn gebaseerd op de frequentie van de betreffende trace:

$$a_p^1(\mathcal{A}^1) = \frac{\sum_{\sigma \in S} \omega(\sigma) \cdot |e_x(\sigma)|}{\sum_{\sigma \in S} \omega(\sigma) \cdot |a_v(\sigma)|}$$

Waarbij:

$S$ = set van staten van alignment automaton $\mathcal{A}^1$

$a_v(\sigma)$ = de beschikbare activiteiten in het model

$e_x(\sigma)$ = de executed actions in de log

$e_x(\sigma) \subseteq a_v(\sigma)$ = de set van executed actions van een bepaalde state is altijd een subset van alle beschikbare activiteiten volgens het model

$\omega(\sigma)$ = het gewicht van elke staat met betrekking tot zijn belangrijkheid voor het berekenen van de precision, gebaseerd op de frequentie van het gebruikte trace

De beschikbare activiteiten zijn die activiteiten die als directe opvolger uitgevoerd kunnen worden volgens het model. De executed actions zijn die activiteiten die ook daadwerkelijk werden uitgevoerd in de log. Deze metriek gaat dus het aantal beschikbare activiteiten vergelijken met het aantal uitgevoerde activiteiten, gewogen op basis van hun belangrijkheid.

Als alle executed actions, ook voorkomen in de log, dan zal deze metriek 1 bedragen wat wijst op een perfecte precision. Hoe minder executed actions in de log voorkomen, hoe lager de $a_p^1$ zal

bedragen. Nochtans zal deze metriek nooit 0 bedragen omdat er altijd enkele uitgevoerde activiteiten in de log zullen zijn.

### 2.3.5 Best Align Precision

Hetzelfde idee als bij de $a_p^1$ kan gebruikt worden om een metriek op te stellen die rekening houdt met alle optimale alignments ($\mathcal{A}$) van een trace. Het enige verschil zit in de bepaling van het gewicht ω. Er wordt nu voor de belangrijkheid niet uitsluitend gefocust op de frequentie, maar deze waarde zal op een gewogen manier bepaald worden. Er wordt namelijk rekening gehouden met alle optimale alignments tussen de event log en het procesmodel, waardoor er een meer complete schets van het gedrag in de event log wordt beschouwd. Daarvoor wordt de volgende formule gebruikt [27]:

$$\omega(s) = \sum_{\sigma_L \in L} \frac{\left|\left\{\gamma \in \Gamma^o_{\sigma_L,M} \mid sel_2(\gamma)_{\downarrow A_M} = s \cdot \sigma' \wedge \sigma' \in A_M *\right\}\right|}{|\Gamma^o_{\sigma_L,M}|}$$

Waarbij:

$L$ = een log over $A_L$

$M$ = een model, $M \subseteq A_M *$

$\mathcal{A}$ = een alignment automaton opgebouwd uit alle optimale alignments tussen $L$ en $M$

$\Gamma^o_{\sigma_L,M}$ = de set van alle optimale alignments

$sel_2(\gamma)_{\downarrow A_M}$ = de projectie van het tweede element van elke optimale alignment

$s$ = de staat in alignment automaton $\mathcal{A}$

De formule telt voor elke case het aantal optimale alignments waar de beschouwde staat in voorkomt. Vervolgens wordt dit gedeeld door het totale aantal optimale alignments. Ook deze metriek zal een waarde bekomen die ligt tussen 0 en 1, waarbij de interpretatie dezelfde is als bij de $a_p^1$. Uit het onderzoek in [27] blijkt eveneens dat de waarde voor de $a_p^1$ een goede benadering is voor de Best Align Precision ($a_p$), terwijl de $a_p^1$ veel minder rekentijd nodig heeft dan de $a_p$.

## 2.4 Metrieken gebruikt voor Generalization

### 2.4.1 Behavioral Generalization en Weighted Behavioral Generalization

Ook voor deze metrieken wordt het concept van gewogen artificiële negatieve events toegepast [24]. Door het vergelijken van alle negatieve events die een bepaald positief event voorgaan met het volledige activiteiten alfabet, is het mogelijk om alternatieve events af te leiden die ook toegelaten zouden moeten zijn door het procesmodel na het uitvoeren van de positieve events. Er wordt dus gezocht naar activiteiten die, gegeven een bepaalde sequentie van events, ook toegelaten zouden moeten zijn in het model. Deze events worden *generalized events* genoemd. Deze techniek kan gebruikt worden om te onderzoeken of deze afgeleide generalized events daadwerkelijk aanvaard worden of niet. [12]

De formule die vervolgens voorgesteld wordt in [12] om Weighted Behavioral Generalization te berekenen is:

$$g_B^W = \frac{AG}{AG + DG}$$

Waarbij:

$AG$ = Allowed Generalizations, generalized events die zonder fout konden gereplayed worden en de mogelijkheid van het model om te generaliseren bevestigen

$DG$ = Disallowed Generalizations, generalized events die niet konden herhaald worden door het procesmodel

De waardes van $AG$ en $DG$ worden als volgt berekend: er wordt gestart van een bepaalde bereikte staat in de trace replay. Er wordt vervolgens voor elk negatief event $n \in NE(\sigma_i)$ nagegaan of deze geactiveerd kan worden door het procesmodel. Als dit het geval is wordt $AG$ vermeerderd met $1 - Weight(n)$. Als het procesmodel niet in staat is het negatieve event te parsen, wordt $DG$ vermeerderd met $1 - Weight(n)$. Belangrijk is om op te merken dat deze analyse wordt gemaakt zonder het daadwerkelijk uitvoeren van de negatieve events. Het gewicht van een negatief event schommelt tussen 0 en 1 zoals uitgelegd in sectie 2.3.1, waarbij 1 betekent dat het respectievelijke negatieve event compleet onbruikbaar is om de generalization van een procesmodel te beoordelen. Terwijl een negatief event waarbij het gewicht 0 bedraagt, volledig geschikt is om de capaciteit van een model om te generaliseren te bepalen. Hoe lager het gewicht van een negatief event, hoe hoger de kans is dat, gegeven een activiteitenprefix, naast de geobserveerde activiteiten ook andere activiteiten mogelijk zijn. Met andere woorden hoe beter bruikbaar dit negatief is om de kwaliteitsdimensie generalization te kwantificeren. [12], [24]

De formule die gebruikt wordt voor de niet gewogen Behavioral Generalization is volledig hetzelfde. Het enige verschil zit in de bepaling van de waardes van $AG$ en $DG$. In dit geval wordt $AG$ vermeerderd met 1 als het negatieve event geparsed kan worden en $DG$ wordt ook met 1 verhoogd indien het negatieve event niet kan uitgevoerd worden. Er wordt dus geen rekening gehouden met de kracht van het event om de generalization van het model te bepalen. [12]

Beide metrieken zullen tussen 0 en 1 liggen, waarbij 1 betekent dat het model goed in staat is om overfitting te vermijden en 0 betekent dat het model niet in staat is om onbekend gedrag in het systeem te beschrijven. Een model dat overfitting[2] is, is een model dat enkel gedrag toelaat dat daadwerkelijk geobserveerd is [21]. Hoe meer negatieve events er onbruikbaar ($Weight(n) = 1$) zijn om de generalization te berekenen, hoe lager de Weighted Behavioral Generalization zal zijn.

### 2.4.2 Alignment Based Probalistic Generalization

Om generalization tussen de log en het model te kwantificeren in deze laatste metriek, wordt er gekeken naar elke staat van het model. Gegeven een event $e$ dat voorkomt in staat $s$ van het model,

[2] Het is belangrijk om fitness niet te verwarren met over- of underfitting. Zowel een model dat overfitting is als een model dat underfitting is kan een fitness van 1 hebben.

kunnen alle events $e'$ in de gealigneerde log gevonden worden die voorkomen in diezelfde staat. Elk event $e'$ kan beschouwd worden als een occurrence van een activiteit in staat $s$. Veronderstel dat de staat $s$, $n$ keer bezocht wordt en er $w$ verschillende activiteiten in staat $s$ zijn. Als $n$ groot is en $w$ zeer klein, dan is het onwaarschijnlijk dat een nieuw event, die voorkomt in de staat, zal overeenkomen met een activiteit die nog niet eerder gezien werd in $s$. Nochtans, als $n$ en $w$ in dezelfde grootteorder zitten, dan is het meer waarschijnlijk dat een nieuw event dat voorkomt in $s$, zal overeenkomen met een activiteit nog niet eerder gezien in $s$. Generalization kan dus als volgt gekwantificeerd worden, zoals beschreven in [28]:

$$1 - \sum_{s \in S} \frac{\sum_{(\sigma_1, a, \sigma_2) \in split(L)} state\big((\sigma_1, a, \sigma_2)\big)(s)}{|split(L)|} \cdot pnew(seen(L, s), visit(L, s))$$

Waarbij:

$seen(L, s) = |\{a | (\sigma_1, a, \sigma_2) \in split(L) \wedge state\big((\sigma_1, a, \sigma_2)\big)(s) > 0\}|$, de activiteiten die mogelijk gezien worden voor staat s

$visit(L, s) = |\big[(\sigma_1, a, \sigma_2) \in split(L) \wedge state\big((\sigma_1, a, \sigma_2)\big)(s) > 0\big]|$ , het aantal keer dat state s (mogelijks) bezocht werd

$pnew(w, n): \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{P}$ is een functie die de geschatte kans terug geeft dat het volgende event dat s bezoekt overeenkomt met een activiteit die voorheen nog niet werd geobserveerd

Voor alle $w, n \in \mathbb{N} : pnew(w, n) = \frac{w(w+1)}{n(n-1)}$ $als\ n \geq w + 2$ $en\ pnew(wn, n) = 1$ $elders$

De generalization waarde van een model in vergelijking tot een event log zal dicht bij 0 liggen als het zeer waarschijnlijk is dat een nieuwe trace, nieuw gedrag zal bevatten dat nog niet eerder geobserveerd werd. Omgekeerd, de waarde zal kort bij 1 liggen als het zeer onwaarschijnlijk is dat een nieuwe trace, nieuw gedrag zal vertonen. [28]

## 2.5 Conclusie

Op basis van bovenstaande literatuurstudie over de verschillende metrieken die de verschillende kwaliteitsdimensies trachten te kwantificeren, is het moeilijk op te maken hoe de metrieken zich ten opzichte van elkaar verhouden. Elke metriek probeert op een andere manier de respectievelijke kwaliteitsdimensie te kwantificeren. De één doet dit aan de hand van het tellen van de tokens, de ander gaat eerst traces aligneren met het model alvorens iets te berekenen enzovoort. Hierdoor kan elke metriek anders reageren op een gegeven situatie. Er is dus niet geweten hoe consistent de metrieken binnen een bepaalde kwaliteitsdimensie zijn.

Verder is het eveneens niet geweten of elke metriek binnen een bepaalde dimensie zich in dezelfde richting beweegt indien er bijvoorbeeld een andere algoritme gebruikt wordt om het model te ontdekken. Er kunnen metrieken zijn die sterker reageren dan andere metrieken op een bepaalde verandering. Het is belangrijk om dit te weten als verschillende modellen met elkaar vergeleken moeten worden op basis van deze metrieken. Als geweten is hoe gevoelig de metrieken zijn aan en hoe ze reageren op bepaalde situaties, zal de interpretatie van de kwaliteit van een bepaald

procesmodel eenvoudiger en betrouwbaarder zijn. Dit is de reden dat er in het vervolg van deze masterproef een analyse wordt uitgevoerd van deze metrieken. Er zal geprobeerd worden om aan de hand van artificiële event logs nieuwe inzichten te verwerven van de geobserveerde metrieken.

Ten slotte valt op dat, op enkele metrieken na, de interpretatie van de gevonden waarden zeer moeilijk is. De formules zijn zodanig complex dat het niet meer mogelijk is om hier een duidelijke interpretatie over te geven. Dit betekent eveneens dat het niet mogelijk is om in te schatten hoe goed of slecht een bekomen waarde is.

# Hoofdstuk 3: Opzet experimenten

Om een grondige analyse uit te voeren van de verschillende metrieken, is een relatief grote hoeveelheid aan data vereist. Aangezien er zeer weinig reële event logs beschikbaar zijn, wordt er gekozen om gebruik te maken van artificiële event logs. Hiervoor worden in totaal twee experimenten opgezet, waarvan de stappen in dit hoofdstuk kort toegelicht zullen worden.

## 3.1 Experiment 1

### 3.1.1 Doel

Uit de literatuur werd reeds duidelijk dat het zeer moeilijk is om te voorspellen hoe bepaalde metrieken reageren op een gegeven situatie of hoe de verschillende metrieken zich verhouden ten opzichte van elkaar. Het is eveneens niet geweten of de metrieken wel degelijk de kwaliteitsdimensie meten die volgens de theorie wordt voorgelegd.

Om dit te onderzoeken zal een eerste experiment worden opgezet, waarin artificiële event logs gegenereerd worden. Er zal in dit experiment gebruik gemaakt worden van zowel eenvoudigere, als complexere en dus meer realistische systemen voor het genereren van de event logs. Op deze manier wordt getracht een zo volledig mogelijk beeld te vormen over hoe de metrieken reageren op een gegeven situatie.

Deze data zal gebruikt worden om verschillende analyses op uit te voeren, om zo de relatie tussen de verschillende metrieken te onderzoeken. Aangezien zeer weinig tot niets over deze problematiek geweten is in de literatuur kunnen zeer moeilijk hypotheses worden opgesteld voor dit gedeelte. Het is daarom een exploratieve analyse waarin enkele aandachtspunten naar boven gehaald worden die in detail onderzocht zullen worden. De resultaten die gevonden worden, zijn dus informatief en bevestigen, noch verwerpen enige hypothese.

### 3.1.2 Opzet

De opzet van dit experiment is gekoppeld aan een bepaalde procedure die bestaat uit verschillende stappen. In de eerste plaats zullen een aantal artificiële modellen opgesteld worden met behulp van een process tree generator. De volgende stap is om deze process trees te simuleren met als resultaat een verzameling van event logs. Deze event logs worden gebruikt om procesmodellen te ontdekken, waarop ten slotte de evaluatiemetrieken berekend worden. Hoe deze datasets gegenereerd worden zal in deze sectie verder verklaard worden. De werking van de gebruikte Process trees zal eveneens uitgelegd worden.

#### 3.1.2.1 Process trees

Zoals reeds eerder vermeld, zijn er veel verschillende notaties om procesmodellen weer te geven. De meeste metrieken vereisen de voorstelling van een procesmodel door middel van een Petri net.

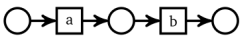
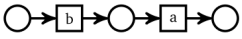
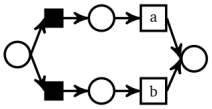
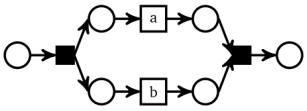
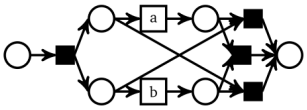
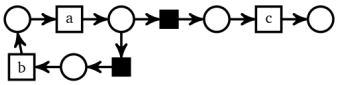
De reden hiervoor is dat Petri nets toelaten om een trace opnieuw af te spelen. Tijdens deze replay wordt de staat van het Petri net weergegeven doormiddel van tokens. Een aantal metrieken vereisen deze eigenschappen van een Petri net, die andere notaties niet hebben. [2]

Voor het opstellen van artificiële event logs, is het echter eenvoudiger om gebruik te maken van Process trees. Er wordt gekozen voor Process trees als notatie binnen de artificiële modelgenerator omwille van twee redenen. Process trees stellen block-structured processen voor die inherent sound zijn. Dit omdat ze er impliciet voor zorgen dat elke control-flow split een corresponderende join heeft van hetzelfde type. Verder laten ze niet toe dat afhankelijkheden of bogen toegevoegd of verwijderd worden tussen de join en split. Ze houden met andere woorden afhankelijkheden lokaal. Dankzij deze eigenschap zal de simulator in een volgende stap nooit kunnen vastlopen. Een tweede reden voor het gebruik van Process trees, is eveneens gekoppeld aan het feit dat ze block-structured processen voorstellen. Hierdoor zijn ze namelijk eenvoudig incrementeel op te bouwen. Er kunnen op een eenvoudige manier blokken vervangen worden zonder dat dit gevolgen met zich meebrengt voor de algemene structuur en correctheid van het model. [22], [29]

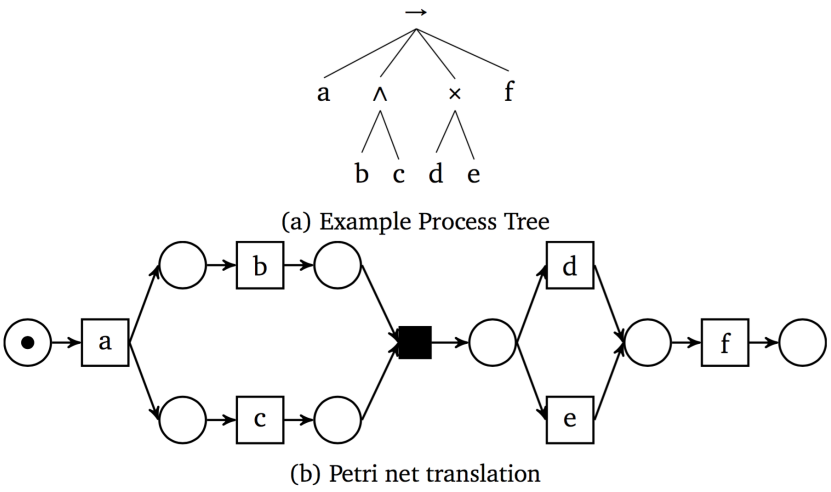
De process trees bevatten de volgende vijf controlflow operatoren zoals gedefinieerd [22]:

- sequence ($\rightarrow$)
- exclusive choice ($\times$)
- parallellisme ($\wedge$)
- or choice ($\vee$)
- loop ($\circlearrowleft$)

In onderstaande figuur worden elk van deze operatoren voorgesteld aan de hand van een vertaling naar een Petri net [22]:

| Process Tree | Petri net | Example traces |
|---|---|---|
| → a b |  | 1 trace: $\{\langle a, b\rangle\}$ |
| ← a b |  | 1 trace: $\{\langle b, a\rangle\}$ |
| × a b |  | 2 traces: $\{\langle a\rangle, \langle b\rangle\}$ |
| ∧ a b |  | All interleavings of the children: $\{\langle a, b\rangle, \langle b, a\rangle\}$ |
| ∨ a b |  | All interleavings, with the option to skip all but one child: $\{\langle a\rangle, \langle b\rangle, \langle a, b\rangle, \langle b, a\rangle\}$ |
| ↻ a b c |  | Infinite number of traces of the pattern $a(ba)^*c$ : $\{\langle a, c\rangle, \langle a, b, a, c\rangle, \langle a, b, a, b, a, c\rangle, \ldots\}$ |

*Figuur 10: Voorstelling van Process tree operatoren*




*Figuur 11: Vertaling van Process tree naar Petri net*

Om een process tree te lezen, moet er van boven naar beneden en van links naar rechts gelezen worden. Dit zal uitgelegd worden aan de hand van het voorbeeld in Figuur 11 [22]. De root node van deze process tree is de sequence operator, zoals weergegeven door het →-symbool. Dit wil zeggen dat alle kinderen van deze operator achtereenvolgens van links naar rechts uitgevoerd moeten

worden. De eerste activiteit die dus uitgevoerd moet worden is activiteit a. Slechts als deze activiteit voltooid is kan het volgende kind uitgevoerd worden. Dit is de parallelle operator (∧-symbool). Deze operator betekent dat zijn kinderen in eender welke volgorde uitgevoerd kunnen worden. Het volgende kind van de root node, is de choice operator (×-symbool) wat betekent dat er gekozen moet worden tussen het uitvoeren van ofwel activiteit d of e. Ten slotte moet activiteit f uitgevoerd worden.

#### 3.1.2.2 Process tree generation

Om een event log te genereren moeten er in de eerste plaats, aan de hand van enkele parameters, verschillende systemen opgesteld. Een systeem verwijst naar het werkelijke gedrag in het proces, of met andere woorden het onderliggende proces. Dit doorgaans onbekend proces, definieert de werkelijke manier waarop het werk gedaan kan worden. De process tree generator vertrekt vanuit een populatie van process trees. De gebruiker karakteriseert deze populatie door de control-flow eigenschappen van de trees te bepalen aan de hand van onderstaande parameters. Uit die populatie wordt vervolgens een willekeurige steekproef, bestaande uit willekeurige process trees, getrokken. Het volledige gebruikte algoritme wordt uitgelegd in [30].

Er wordt zo goed mogelijk gediversifieerd tussen mogelijke controlflow patronen dat een systeem kan hebben op basis van de input parameters. De mogelijke parameters[3] zijn:

- Mode: de modus van het aantal activiteiten
- Min: het minimumaantal activiteiten
- Max: het maximumaantal activiteiten
- Sequence: de kans op een sequentie
- Choice: de kans op een Exclusive Choice operator
- Parallel: de kans op een Parallel operator
- Loop: de kans op een Loop operator
- Or: de kans op een Or operator
- Silent: de kans op een silent/invisible activity
- Duplicate: de kans op duplicate activiteiten
- Long term-dependency[4]

Elk van deze parameters krijgt een bepaalde waarde die elk (met uitzondering van mode, min en max) een kans weergeven dat het respectievelijke construct voorkomt in de process tree. De parameter settings van dit experiment worden weergegeven in de volgende tabellen:

[3] De som van de kans op sequence, choice, parallel, loop en or moet gelijk zijn aan 1.

[4] Gedrag in het proces waarbij keuzes die gemaakt moeten worden, afhangen van keuzes die reeds eerder in het proces gemaakt werden. [22]

| population | mode | min | max | sequence | choice | parallel | loop | or | silent | duplicate | lt-dependancy |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 15 | 10 | 20 | 0,40 | 0,30 | 0,30 | 0,00 | 0,00 | 0,00 | 0,00 | 0,00 |
| 2 | 15 | 10 | 20 | 0,40 | 0,30 | 0,00 | 0,30 | 0,00 | 0,00 | 0,00 | 0,00 |
| 3 | 15 | 10 | 20 | 0,40 | 0,30 | 0,15 | 0,15 | 0,00 | 0,00 | 0,00 | 0,00 |
| 4 | 15 | 10 | 20 | 0,40 | 0,30 | 0,15 | 0,00 | 0,15 | 0,00 | 0,00 | 0,00 |
| 5 | 15 | 10 | 20 | 0,40 | 0,30 | 0,00 | 0,15 | 0,15 | 0,00 | 0,00 | 0,00 |
| 6 | 15 | 10 | 20 | 0,40 | 0,30 | 0,10 | 0,10 | 0,10 | 0,00 | 0,00 | 0,00 |
| 7 | 15 | 10 | 20 | 0,40 | 0,30 | 0,10 | 0,10 | 0,10 | 0,10 | 0,00 | 0,00 |
| 8 | 15 | 10 | 20 | 0,40 | 0,30 | 0,10 | 0,10 | 0,10 | 0,00 | 0,10 | 0,00 |
| 9 | 15 | 10 | 20 | 0,40 | 0,30 | 0,10 | 0,10 | 0,10 | 0,00 | 0,00 | 0,50 |
| 10 | 15 | 10 | 20 | 0,45 | 0,40 | 0,00 | 0,00 | 0,15 | 0,00 | 0,00 | 0,00 |

*Tabel 3: Parameters experiment 1a*

| population | mode | min | max | sequence | choice | parallel | loop | or | silent | duplicate | lt-dependency |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 20 | 15 | 25 | 0,45 | 0,20 | 0,10 | 0,25 | 0,00 | 0,10 | 0,10 | 0,50 |
| 2 | 20 | 15 | 25 | 0,35 | 0,20 | 0,05 | 0,30 | 0,10 | 0,10 | 0,00 | 1,00 |
| 3 | 20 | 15 | 25 | 0,30 | 0,30 | 0,00 | 0,30 | 0,10 | 0,00 | 0,10 | 0,00 |
| 4 | 20 | 15 | 25 | 0,45 | 0,20 | 0,10 | 0,25 | 0,00 | 0,00 | 0,00 | 0,50 |
| 5 | 20 | 15 | 25 | 0,45 | 0,20 | 0,00 | 0,25 | 0,10 | 0,00 | 0,10 | 1,00 |

*Tabel 4: Parameters experiment 1b*

De parameter settings van het tweede gedeelte van experiment 1 worden als meer complex gezien, omdat er onder meer een hogere kans is op een long-term dependency. Hiernaast is de kans op het voorkomen van een loop ook hoger, net als het voorkomen van silent en duplicate tasks. De combinatie van deze factoren, samen met het verhoogde gemiddelde aantal activiteiten, zorgt ervoor dat de modellen die uit deze systemen ontdekt zullen worden complexer zijn.

Voor het gedeelte van de dataset met de eenvoudigste systemen (experiment 1a), wordt er gebruik gemaakt van 10 populaties, waaruit telkens één systeem wordt gehaald. Voor het complexere gedeelte van de dataset (experiment 1b) worden uit vijf populaties telkens twee systemen gehaald. Voor beide delen worden dus 10 systemen opgesteld.

#### 3.1.2.3 Event log generation

De opgestelde process trees worden vervolgens gesimuleerd om zo de event logs te genereren, met elk een bepaald niveau van completeness en noise. Het algoritme dat gebruikt wordt om de event logs te genereren wordt in detail uitgelegd in [30]. Enkel de delen die relevant zijn voor deze masterproef worden in deze sectie uitgelegd.

Completeness stelt voor hoeveel van de mogelijke traces in het systeem, ook voorkomen in de event log. Dit wil zeggen dat als in het systeem een activiteit gevolgd wordt door een andere, dit minstens één keer moet voorkomen in de log om tot een complete log te komen [31]. Om de completeness van een event log te beïnvloeden, werd een algoritme geschreven dat uitrekent hoeveel verschillende traces de betreffende process tree (het systeem) bevat. Op deze manier kan completeness gecontroleerd worden door te specifiëren hoeveel unieke paden (e.g. 75% van het maximumaantal) uit het systeem in de event log mogen zitten [32]. Als in de process trees loops voorkomen, gaat de berekening van het maximumaantal mogelijke paden ervan uit dat deze loop niet oneindig kan voorkomen. In het algoritme werd dit maximumaantal op drie gezet. Deze assumptie wordt ook de

fairness assumption genoemd. Het zorgt ervoor dat oneindig gedrag, dat als onrealistisch gezien wordt, vermeden wordt [33]. Voor dit experiment worden vier niveaus van completeness gehanteerd, namelijk 25, 50, 75 en 100%.

Noise wordt gedefinieerd als incorrect gedrag in de event log. Het wordt toegevoegd aan de event logs, om de realiteit zo goed mogelijk na te bootsen. In event logs uit de praktijk zal er altijd een zeker niveau van noise aanwezig zijn. Het is de taak van de gebruikte discovery algoritmes om te bepalen of het gedrag werkelijk gedrag is uit het systeem, of dat het gaat om noisy gedrag. Het is niet omdat gedrag slechts infrequent voorkomt, dat het ook noisy gedrag is. [34]

Er zijn verschillende types van noise die kunnen toegepast worden op de artificiële event logs [35]. De eerste drie types *missing head*, *body* en *tail* verwijderen respectievelijk sub-traces in het begin, midden of einde van de traces. Als er een trace $\sigma = t_1 \dots t_{n-1} t_n$ veronderstelt wordt, gaat het begin van $t_1$ tot $t_{n/3}$. Het midden gaat van $t_{(n/3)+1}$ tot $t_{(2n/3)}$ en het einde van $t_{(2n/3)+1}$ tot $t_n$. De sub-traces die verwijderd worden bevatten tenminste één activiteit en maximum alle taken in het respectievelijke begin, midden of einde. Een volgende type, namelijk *swap task*, gaat in trace $\sigma$ twee taken van plaats verwisselen. *Remove task* gaat een willekeurige activiteit uit trace $\sigma$ verwijderen. *Mix all*, tot slot, voert willekeurig een van de vijf andere noise types (elke met een gelijke probabiliteit) uit op de betreffende trace. Real-life logs bevatten typisch een mix van deze verschillende types van noise [35].

In dit experiment worden voor het eerste eenvoudigste gedeelte van de dataset vier niveaus van noise gehanteerd, namelijk 0, 5, 10 en 15%. Voor het tweede gedeelte worden eveneens vier niveaus gebruikt, echter ditmaal 0, 25, 50 en 75%. Een log met bijvoorbeeld 5% noise wordt gecreëerd door 5.26% (0.05/0.95) van de traces van een noise-vrije log te selecteren. Aan deze subset van traces worden vervolgens de verschillende types van noise toegevoegd met uitzondering van de *swap task*. Deze noisy traces worden dan opnieuw toegevoegd aan de originele log. De resulterende log heeft bijgevolg n*(1+0.0526) traces van dewelke 0.0526*n, ofwel 5%, traces noisy zijn. Hoe meer noisy traces aanwezig zijn, hoe complexer het uiteindelijke model zal worden. Dit is de reden dat voor het tweede gedeelte van de dataset hogere niveaus van noise gehanteerd werden dan in het het eerste gedeelte.

Voor experiment 1a worden voor elke combinatie van systeem, noise-niveau en completeness-niveau 10 logs gegenereerd. Dit komt neer op 1600 (10*4*4*10) event logs. Dus voor elk systeem worden in totaal 160 event logs gegenereerd. Voor experiment 1b worden voor elke combinatie 5 logs gegenereerd, wat resulteert in 800 (10*4*4*5) event logs. Voor elk systeem worden dus in totaal 80 event logs gegenereerd.

#### 3.1.2.4 Process discovery

De volgende stap is om uit de opgestelde event logs aan de hand van verschillende process discovery algoritmes, modellen te minen waarop uiteindelijk de verschillende metrieken berekend zullen

worden. De gebruikte algoritmes zijn Alpha miner [7], ILP miner [36], Heuristics miner [31], Flower miner en Infrequent Inductive miner [37]. Voor elk van deze algoritmes warden de standaardwaarden gebruikt. De output van deze algoritmes zijn Petri net modellen. Er worden bijgevolg 4000 modellen (800*5) voor experiment 1a en 8000 modellen opgesteld (1600*5) voor experiment 1b. Deze stap wordt uitgevoerd met behulp van de Vlaamse Super Computer (VSC) en ProM.

De voorlaatste stap is om van elk van deze modellen de metrieken te berekenen. In totaal worden negen metrieken berekend die elk besproken werden in hoofdstuk 2, namelijk:

- Alignment Based Fitness
- Token Based Fitness
- Behavioral Recall
- Alignment Based Generalization
- Behavioral Generalization
- Alignment Based Precision
- Behavioral Precision
- Best Align Precision
- One Align Precision

Er wordt dus gebruik gemaakt van in totaal drie fitness metrieken, twee generalization metrieken en vier precision metrieken. Dit komt neer op in totaal 36000 observaties (4000*9) voor experiment 1a en 72000 observaties (8000*9) voor experiment 1b. Ook bij deze stap wordt gebruik gemaakt van de VSC en CoBeFra [11].

Tot slot zal er een statistische analyse worden uitgevoerd op de gegenereerde data. Dit zal worden uitgevoerd met behulp van R. Er zal getracht worden om visuele voorstellingen te geven van de verschillende analyses die uitgevoerd worden, om zo een duidelijk beeld te schetsen van de resultaten.

### 3.1.3 Ontbrekende en negatieve waarden

In elk van de twee delen van dit experiment konden bepaalde waarden niet berekend worden vanwege beperkingen in reken- en geheugencapaciteit. De berekening werd uitgevoerd met behulp van het VSC (Vlaams Supercomputer Centrum), gefinancierd door het FWO (Fonds voor Wetenschappelijk onderzoek). De berekeningen werden elk uitgevoerd met maximum 15gb geheugen en gebruikmakend van 2.8 GHz CPU’s. De ontbrekende waarden worden samengevat in de volgende tabel:

| Metriek | Experiment 1a (totaal 4000) | % | Experiment 1b (totaal 8000) | % | Totaal (12000) | % |
|---|---|---|---|---|---|---|
| Alignment Based Fitness | 844 | 21,1 | 5763 | 72,04 | 6607 | 55,06 |
| Alignment Based Precision | 4 | 0,1 | 5556 | 69,45 | 5560 | 46,33 |
| Best Align Precision | 990 | 24,75 | 7473 | 93,41 | 8463 | 70,52 |
| Behavioral Generalization | 6 | 0,15 | 0 | 0 | 6 | 0,05 |
| Behavioral Precision | 274 | 6,85 | 0 | 0 | 274 | 2,28 |
| Behavioral Recall | 0 | 0 | 0 | 0 | 0 | 0 |
| Token Based Fitness | 353 | 8,83 | 1403 | 17,54 | 1756 | 14,63 |
| Alignment Based Generalization | 28 | 0,7 | 5753 | 71,91 | 5781 | 48,18 |
| One Align Precision | 567 | 14,18 | 4971 | 62,14 | 5538 | 46,15 |
| **Totaal** | 3066 | 8,5 | 30919 | 42,9 | 33985 | 31,47 |

*Tabel 5: Samenvatting ontbrekende waarden*

Er zijn in totaal 3066 (8,5%) ontbrekende waarden in experiment 1a, verdeeld over zes metrieken. Experiment 1b bevat in totaal 30919 (42,9%) ontbrekende waarden, verdeeld over acht metrieken. Deze ontbrekende waarden zijn, zoals eerder vermeld, het gevolg van computationele beperkingen in geheugen en/of tijd. Als de supercomputer zijn capaciteit in geheugen en/of tijd heeft bereikt, stopt deze met het berekenen van de metrieken die op dat moment nog niet voltooid zijn. Er wordt dus geen uitkomst gevonden voor die betreffende metrieken.

De reden dat bij experiment 1a aanzienlijk minder ontbrekende waarden aanwezig zijn, relateert aan het feit dat bij experiment 1a eenvoudigere modellen gebruikt werden. De algoritmes om de metrieken te bereken hebben hierdoor minder moeite om tot een oplossing te komen binnen de vooropgestelde tijd en geheugen.

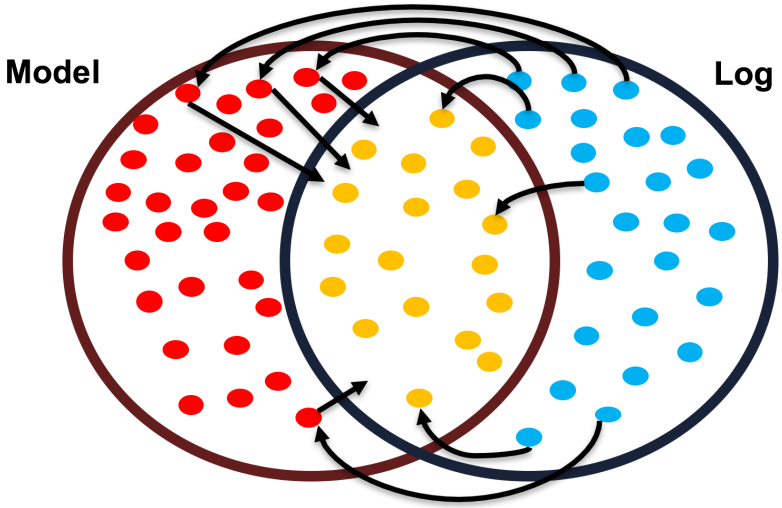
Opvallend is dat de Behavioral Based metrieken in het geval van complexe modellen minder moeite hebben om tot een uitkomst te komen dan de andere metrieken. Dit kan gezien worden als een sterkte van dit type van metrieken. Het zijn voornamelijk de Alignment Based metrieken die het meeste moeite hebben op gebied van geheugen en tijd.

## 3.2 Experiment 2

### 3.2.1 Doel

Het doel van het tweede experiment dat uitgevoerd zal worden in deze masterproef werd eveneens duidelijk uit de literatuurstudie, namelijk dat fitness en precision onafhankelijk zouden moeten zijn van elkaar. Deze stelling vloeit voort uit de definities van model-log fitness en precision die opgesteld werden met behulp van het Venn diagram in sectie 2.1, ditmaal enkel met het model en de event log[5], kan deze stelling geïllustreerd worden.

[5] Het systeem heeft geen invloed op de berekening van de model-log fitness en precision, dus het wordt hier niet in beschouwing genomen.

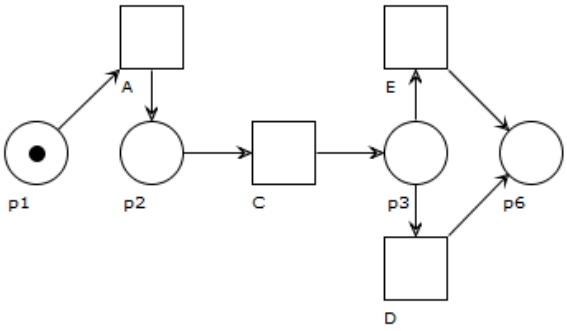



*Figuur 12: Venn diagram (model en log)*

$$Model - \log precision = \frac{|L \cap M|}{|M|} \qquad Model - \log fitness = \frac{|L \cap M|}{|L|}$$

De formules voor deze dimensies delen enkel een gelijke teller die de doorsnede weergeeft van het model met de log (🟡). Dit zijn met andere woorden de traces die zowel voorkomen in de log als het model. Deze term wordt gedeeld door het aantal traces in het model en de log voor respectievelijk precision en fitness. Als bijgevolg een andere log wordt beschouwd waarin bijvoorbeeld een grotere hoeveelheid traces aanwezig zijn, maar het aantal overeenkomende traces gelijk blijft, zou de precision niet mogen veranderen. Omgekeerd, als voor een bepaalde log aan de hand van twee discovery algoritmes verschillende modellen ontdekt worden die beiden hetzelfde aantal fitting traces hebben, zou de fitness voor beiden hetzelfde moeten zijn. In het algemeen kan gezegd worden dat de hoeveelheid non-fitting cases in het model of de log geen invloed mag hebben op respectievelijk de fitness of precision.

Om deze stelling verder te verklaren kan het volgende voorbeeld beschouwd worden. Stel dat met behulp van log $L_1 = [\langle a, c, e \rangle, \langle b, c, d \rangle]$, het volgende model ontdekt kan worden:



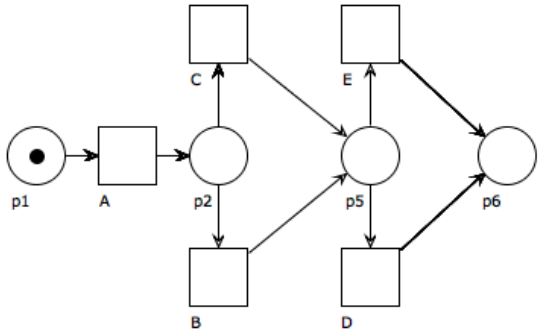

*Figuur 13: Model M1*

Zowel de log als het model bevatten twee mogelijke traces, waarvan één fittend is (namelijk de eerste van $L_1$). De model-log fitness en precision bedragen vervolgens op basis van de definitie van het Venn diagram beiden 0,5 (1/2). Er zullen nu vier situaties uitgewerkt worden, waarin telkens ofwel de log ofwel het model constant worden gehouden. De situaties zijn de volgende:

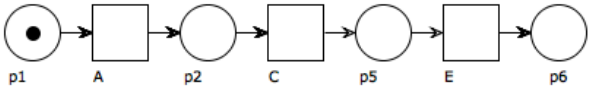
1. precision daalt zonder dat de fitness veranderd
2. precision stijgt zonder dat de fitness veranderd
3. fitness daalt zonder dat de precision veranderd
4. fitness stijgt zonder dat de precision veranderd

Voor situatie 1 en 2 wordt de log constant gehouden worden, terwijl voor situatie 3 en 4 het model constant wordt gehouden. Er wordt dus voor de eerste twee situaties respectievelijk traces toegevoegd en verwijderd in het rode gebied uit Figuur 12. Voor de derde en vierde situatie, worden respectievelijk traces toegevoegd en verwijderd in het blauwe gebied uit Figuur 12.

Stel dat voor situatie 1 model M2 (Figuur 14) wordt ontdekt (bijvoorbeeld met behulp van een ander discovery algoritme). In dit geval bevat het model vier mogelijke traces, terwijl er nog steeds slechts één trace uit $L_1$ op dit model correct kan worden afgespeeld. Hierdoor daalt de model-log precision tot 0,25 (1/4) terwijl de model-log fitness onveranderd blijft. Voor situatie 2 wordt model M3 (Figuur 15) beschouwd, een model dat slechts één trace bevat, die fittend is met de log. Hierdoor stijgt de model-log precision naar 1 (1/1) terwijl dat de model-log fitness nog steeds 0,5 bedraagt.




*Figuur 14: Model M2*

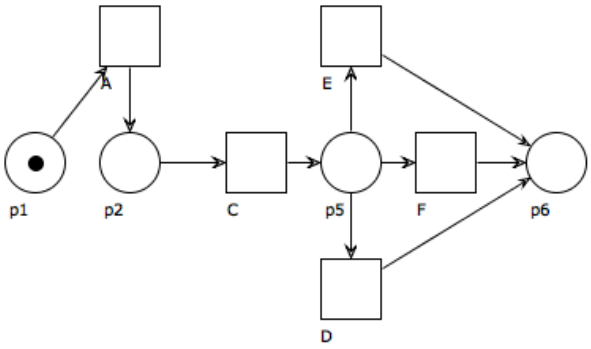



*Figuur 15: Model M3*

Voor situatie 3 en 4 wordt opnieuw model M1 beschouwd, die voor deze twee situaties als een constante wordt gezien. Stel vervolgens dat voor situatie 3 de volgende event log werd gebruikt: $L_2 = [\langle a, c, e\rangle, \langle b, c, d\rangle, \langle b, c, e\rangle]$. Nu bedraagt de model-log fitness 0,33 (1/3) en de model-log precision nog steeds 0,5. In de laatste situatie zou event log $L_3 = [\langle a, c, e\rangle]$ beschouwd kunnen worden, waarbij de model-log precision nog steeds gelijk is gebleven, maar waar de model-log precision nu gestegen is naar 1 (1/1).

Aan de hand van dit voorbeeld wordt aangeduid dat het mogelijk is dat de fitness en precision onafhankelijk van de andere kunnen stijgen of dalen zolang het aantal fittende cases constant blijft. Met andere woorden, door het toevoegen of verwijderen van non-fitting cases in ofwel het model, ofwel de event log is het onmogelijk om zowel de precision als de fitness te doen stijgen of dalen. Dit is enkel mogelijk indien een trace wordt toegevoegd of verwijderd in het gele gebied uit Figuur 12. Als opnieuw het originele voorbeeld beschouwd wordt, kan dit aangetoond worden. Ditmaal wordt

model M4 (Figuur 16) beschouwd in combinatie met $L_4 = [\langle a, c, e \rangle, \langle b, c, d \rangle, \langle a, c, f \rangle]$. Er werden in zowel het model als de event log een fitting trace toegevoegd (namelijk $\langle a, c, f \rangle$). Hierdoor verhogen zowel de model-log precision als de model-log fitness tot 0,66 (2/3).




*Figuur 16: Model M4*

In tegenstelling tot de theorie zijn er in de praktijk enkele precision-metrieken die deze onafhankelijkheid blijken te negeren, namelijk bij de One Align en Best Align Precision. Deze metrieken gaan namelijk eerst de traces aligneren met het model en daardoor non-fitting traces, die op een eerder moment niet in rekening werden genomen, toch opnemen in de berekening. Om dit te illustreren met behulp van bovenstaande Figuur 12: door het aligneren van de traces met het model worden traces die zich in blauwe gebied bevinden, gealigneerd met traces uit het rode of gele gebied (het model). Dus als een non-fitting trace gealigneerd wordt met een eerder niet geobserveerde trace, wordt een trace in het rode gebied veranderd in een trace in het gele gebied. Als het daarentegen gealigneerd wordt met een trace die reeds geobserveerd werd (dus uit het gele gebied), veranderd er niets (met uitzondering van een verandering in frequentie voor die trace). Door deze procedure verdwijnt het blauwe en rode gebied en vergroot het gele gebied.

Als aan de oorspronkelijke verzameling van non-fitting traces in de log, nog meer non-fitting traces worden toegevoegd, zou dit theoretisch gezien geen effect mogen hebben op het niveau van de precision. Nochtans, als met deze analyse in het achterhoofd opnieuw gekeken wordt naar de formule voor One Align en Best Align Precision, wordt duidelijk dat bij het berekenen van deze metrieken de teller van de model-log precision toeneemt terwijl de noemer gelijk zal blijven (aangezien het model ongewijzigd blijft). Met andere woorden, de model-log precision stijgt waardoor deze alignment-based metrieken een vertekend en positiever beeld kunnen geven over de werkelijke waarde van precision.

In dit experiment zal nagegaan worden welke invloed de non-fitting cases hebben op de verschillende metrieken. Er zal niet enkel gekeken worden naar de One Align en Best Align Precision, maar ook naar de andere precision-, fitness- en generalization-metrieken. Er wordt verwacht dat de One Align en Best Align Precision, maar ook de Alignment Based Precision, toenemen naarmate het niveau van non-fitting cases toeneemt. Dit zal beschouwd worden als de nulhypothese. Verder wordt eveneens verwacht dat de fitness-metrieken dalen met een toenemend niveau van non-fitting cases, wat ook uit de theorie kan opgemaakt worden. Hoe de andere metrieken gaan reageren is onduidelijk uit de literatuur.

### 3.2.2 Opzet

Voor dit experiment worden er op dezelfde manier process trees gegenereerd als bij het eerste experiment. In totaal worden tien process trees opgesteld, met elk verschillende parameters. Deze parameters zijn dezelfde als die van experiment 1a en zijn dus terug te vinden in Tabel 3.

Deze process trees worden vervolgens gebruikt om Petri nets te genereren, waardoor het systeem als model gebruikt kan worden. De verklaring hiervoor is dat er een artificieel systeem wordt opgezet aan de hand van de eerder uitgelegde parameters, waarmee process trees worden opgesteld. Deze worden op hun beurt gebruikt om Petri nets te genereren. De process tree stelt dus als het ware het systeem voor. In de realiteit is het systeem doorgaans onbekend, maar aangezien in deze experimenten gebruik gemaakt wordt van process trees, kunnen deze gebruikt worden als systemen.

Uit deze process trees worden event logs gegenereerd, gebruikmakend van hetzelfde algoritme als bij experiment 1. Er worden opnieuw vier niveaus van completeness toegepast, namelijk 25, 50, 75 en 100%. Het verschil met het eerste experiment is dat er ditmaal voor elk systeem vijf event logs worden gegeneerd zonder noise. Hierdoor zouden de opgestelde logs perfect moeten fitten met het model. Op dit moment zijn dus 200 (10*4*5) event logs gegenereerd, die elk noise-free zijn.

Vervolgens worden op basis van deze noise-free logs negen logs opgesteld, waaraan telkens een verschillend niveau van non-fitting cases wordt toegevoegd. Dit wordt bereikt door een bepaalde hoeveelheid noise toe te voegen (zoals besproken in sectie 3.1.2.3) aan de traces van de noise-free log, zodat deze traces niet meer fitten met het model. Het is noodzakelijk om ervoor te zorgen dat de traces waar noise aan toegevoegd wordt, niet per toeval fitten met het systeem/model. Als dit het geval zou zijn, kan de precision theoretisch gezien terecht toenemen. Het percentage non-fitting cases dat uiteindelijk daadwerkelijk wordt toegevoegd aan de event logs varieert tussen 0% (de originele logs) en 45%. Dit resulteert in totaal tien event logs per systeem en completeness-niveau, wat neer komt op 2000 (200 * 10) event logs.

Vervolgens worden voor elke log de negen metrieken berekend, die ook gebruikt werden in experiment 1. Op deze manier worden dus 18000 (9*2000) observaties bekomen. Merk op dat het niveau van non-fitting cases kan vergeleken worden met de naïeve fitness [2]. Als bijvoorbeeld 25% non-fitting cases aanwezig zijn in de event log, komt dit neer op een naïeve fitness van 75%. Dit wil zeggen dat 75% van de cases correct kunnen worden afgespeeld op het model.

### 3.2.3 Ontbrekende waarden

Ook bij dit experiment komen ontbrekende waarden voor. Deze worden geplot in Figuur 17. Hieruit wordt duidelijk dat naarmate het niveau van non-fitting cases toeneemt, en dus de grootte van de log, het aantal ontbrekende waarden stijgt. Voornamelijk bij Alignment Based Precision en Generalization komen er bij 35, 40 en 45% non-fitting cases zeer veel ontbrekende waarden voor. Bij de twee hoogste niveaus, waren deze metrieken zelfs bij geen enkele berekening in staat om tot een oplossing te komen. De andere weergegeven metrieken bevatten ook ontbrekende waarden,

maar in mindere mate. Er zijn twee metrieken die niet opgenomen werden in de figuur, namelijk Best Align Precision en Behavioral Recall, omdat deze geen ontbrekende waarden bevatten. Dit wijst erop dat zowel de Alignment Based Precision als de Alignment Based Generalization veel moeite hebben met het komen tot een oplossing in geval van grotere logs. De andere metrieken lijken hier minder moeite te hebben om tot een oplossing te komen binnen de opgegeven hoeveelheid tijd en geheugen.


*Figuur 17: Ontbrekende waarden per metriek, per level non-fitting cases*

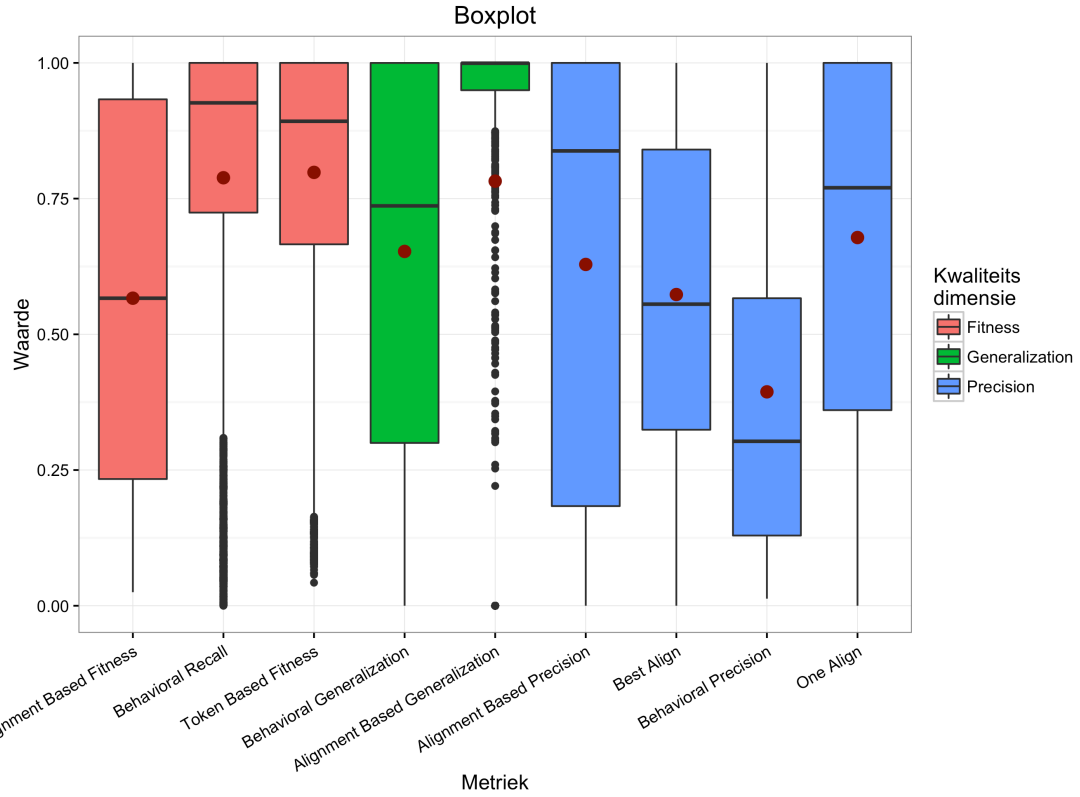
# Hoofdstuk 4: Analyse experiment 1

In dit hoofdstuk zullen verschillende analyses voor experiment 1 uitgevoerd en uitgelegd worden. In de eerste plaats zal er aan de hand van een samenvattende analyse een algemeen beeld gevormd worden van de gebruikte data. Vervolgens zal er aan de hand van een correlatieanalyse gekeken worden of er metrieken zijn die aan elkaar gerelateerd zijn. Een factoranalyse, ten slotte, zal proberen meer inzicht te geven over welke metrieken eenzelfde dimensie meten.

## 4.1 Analyse experiment 1

### 4.1.1 Samenvattende analyse

Om een algemeen beeld te schetsen van de data gebruikt voor experiment 1, wordt in de eerste plaats gebruik gemaakt van een boxplot. Op deze manier kan bijvoorbeeld gekeken worden of metrieken die één bepaalde kwaliteitsdimensie meten, gemiddeld gezien ook dezelfde waarde bekomen.




*Figuur 18: Boxplot voor experiment 1*

Uit bovenstaande figuur met boxplots voor elk van de geobserveerde metrieken in experiment 1 blijkt dat er een redelijke spreiding is tussen de verschillende metrieken. Zo ligt het gemiddelde (weergegeven door de rode bollen) van de Alignment Based Fitness buiten de boxplots van de andere twee fitness-metrieken. Deze twee metrieken liggen echter wel relatief kort bij elkaar. De gemiddelden van de metrieken voor de andere twee dimensies liggen eveneens relatief verspreid. Wat eveneens opvalt is dat de spreiding binnen één metriek sterk verschillend is. Zo is de spreiding van de Behavioral Generalization relatief hoog, terwijl die van de Alignment Based Generalization zeer laag is.

### 4.1.2 Correlatieanalyse

Om eventuele verbanden tussen de verschillende metrieken te onderzoeken, kan er in de eerste plaats een correlatieanalyse uitgevoerd worden. Aan de hand van deze analyse kan gekeken worden of er metrieken zijn die gecorreleerd zijn aan elkaar, zowel positief als negatief. Aan de hand van een visuele weergave wordt getracht groepen van metrieken te vinden die sterk gecorreleerd zijn met elkaar en zich dus gelijkaardig gedragen.

De correlatiematrix voor experiment 1 ziet er als volgt uit:


*Figuur 19: Correlatiematrix van experiment 1*

Hoe donkerder de vakjes, hoe sterker de twee metrieken met elkaar gecorreleerd zijn. Uit deze correlatiematrix zijn twee duidelijke groepen op te maken van sterk positief correlerende metrieken. De eerste groep bestaat uit de drie geobserveerde fitness-metrieken (Token Based Fitness, Behavioral Recall en Alignment Based Fitness), maar ook Behavioral Generalization correleert sterk met deze groep. Dit is een eerste aanwijzing dat deze metriek niet de dimensie generalization meet, maar eerder de dimensie fitness. De tweede groep bestaat uit de vier geobserveerde precision-metrieken (One Align, Behavioral, Best Align en Alignment Based Precision). Wat vervolgens opvalt is dat deze twee groepen zijn, welsiwaar relatief licht, negatief gecorreleerd zijn aan elkaar. Dit zou er op kunnen wijzen dat de dimensies die deze metrieken meten, eveneens negatief gecorreleerd zijn aan elkaar, met andere woorden dat fitness en precision negatief gecorreleerd zijn. De laatste,

nog niet besproken metriek, is Alignment Based Generalization. Deze metriek blijkt met geen enkele andere metriek te correleren en meet dus volgens deze matrix een andere dimensie, die niet door de rest van de metrieken gemeten wordt.

### 4.1.3 Factoranalyse

De reden dat gekozen wordt voor een factoranalyse is omdat elke metriek één bepaalde kwaliteitsdimensie meet. Uit de literatuurstudie kan gesteld worden dat alle metrieken die eenzelfde dimensie meten, zich hetzelfde gaan gedragen naargelang het procesmodel en de eventlog waarop de metriek berekend wordt. Met andere woorden, metrieken binnen één kwaliteitsdimensie zouden zich allemaal in eenzelfde richting moeten bewegen en niet tegengesteld.

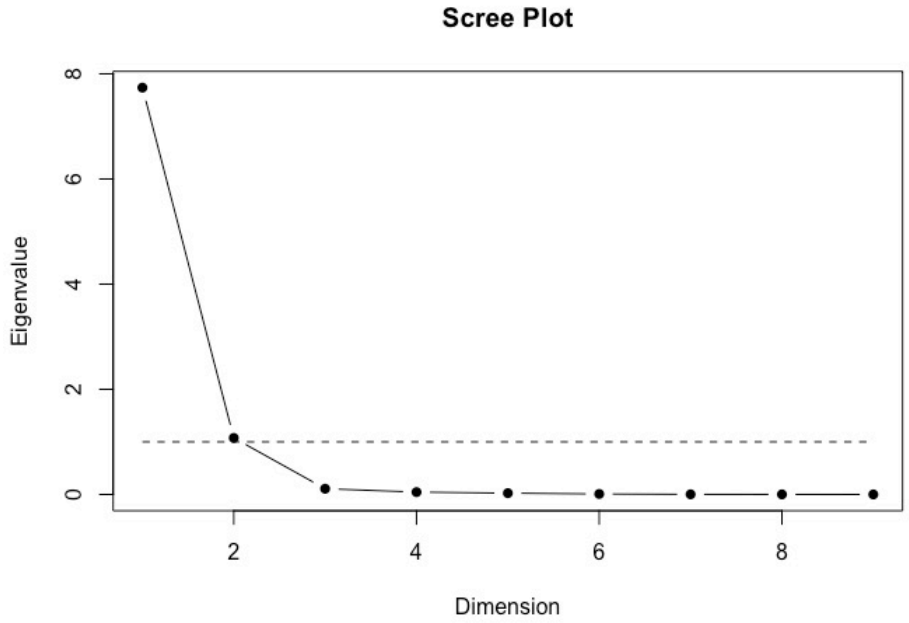
Een factoranalyse gaat trachten de metrieken te groeperen naargelang hun gedrag in de data. Het probeert om de onderliggende structuur tussen de variabelen te definiëren [38]. Als dit het geval is, zouden drie duidelijke groepen van metrieken moeten bekomen worden na het uitvoeren van de factoranalyse, namelijk de drie geobserveerde kwaliteitsdimensies (fitness, precision en generalization).

Er wordt niet vertrokken vanuit een vooropgesteld model, dat bestaat uit de drie geobserveerde kwaliteitsdimensies met hun respectievelijke metrieken. Er wordt dus een exploratieve factoranalyse (EFA) uitgevoerd op de data [39]. Meer specifiek, er wordt gezocht naar de relatie tussen de variabelen (de metrieken) om zo groepen van variabelen te identificeren die latente variabelen (factoren) gaan vormen. Deze analyse wordt eveneens de R factoranalyse genoemd [38]. De reden dat gekozen wordt voor een EFA en niet voor een Confirmatorische Factoranalyse (CFA) is om ervoor te zorgen dat eventuele andere dimensies die naar voor kunnen komen tijdens de analyse niet over het hoofd gezien worden. Bij een CFA wordt namelijk enkel getest of de gemeten variabelen een kleinere groep van constructen meten. Om een CFA uit te voeren, moet de onderzoeker specifiëren hoeveel factoren er bestaan voor een bepaalde set aan variabelen en op welke factoren de variabelen een hoge lading zullen hebben alvorens het resultaat berekend kan worden [38]. Omdat in deze masterproef net onderzocht wordt of er metrieken zijn die eventueel ook andere dimensies meten, of metrieken die zich anders gedragen dan de rest van de metrieken in de betreffende dimensie heeft het dus geen zin om een CFA uit te voeren. In het vervolg van deze masterproef zal naar de EFA vernoemd worden als factoranalyse.

Om de factoranalyse uit te voeren, mogen geen ontbrekende waarden aanwezig zijn in de dataset. Nochtans bevat de dataset van experiment 1, zoals eerder besproken, veel ontbrekende waarden, doordat problemen voorkwamen tijdens de berekening in verband met het geheugen en/of de tijd. Daarom worden enkel die observaties geselecteerd waarvoor alle 9 metrieken correct berekend werden. In totaal zijn er, na het verwijderen van de niet complete cases, nog 2947 cases over. We beschikken in totaal dus over 26523 bruikbare observaties (9 metrieken per case).

#### 4.1.3.1 Aantal factoren

In de eerste plaats moet bepaald worden hoeveel factoren opgesteld moeten worden. Dit kan gedaan worden aan de hand van een screeplot. Er zijn echter meerdere methoden om te bepalen hoeveel factoren gebruikt moeten worden op basis van deze scree plot. Een eerste is om te kijken naar die factoren die een eigenwaarde hebben hoger dan één [38], [40]. Voor dit experiment zou op basis hiervan gekozen worden voor twee factoren.




*Figuur 20: Scree plot experiment 1*

Een tweede manier is om te kijken naar waar de daling in de curve een knik maakt [38]. Alle factoren die na de knik vallen, kunnen achterwege gelaten worden omdat er dan door een toevoeging van een factor niet veel meer extra variantie verklaard wordt. Voor dit experiment zou op basis van deze redenering gekozen worden voor drie factoren.

Er wordt dus gekozen om zowel een twee- als een drie-factoranalyse uit te voeren. De resultaten van deze analyses kunnen teruggevonden worden in bijlage, waar ook een samenvatting staat van de kwaliteit van deze analyses. Er wordt uiteindelijk op basis van de kwaliteit en de interpreteerbaarheid van de factoren gekozen voor de twee-factoranalyse.

#### 4.1.3.2 Kwaliteit factoranalyse

Alvorens conclusies genomen kunnen worden, moet de kwaliteit van de factoranalyse beoordeeld worden. Deze kwaliteit wordt weergegeven aan de hand van verschillende statistieken, testen en maatstaven die samengevat worden in onderstaande tabel.

| | Communaliteit | |
|---|---|---|
| **Metriek** | 2 Factoren | **MSA** |
| Alignment Based Fitness | 0,70254 | 0,80018 |
| Alignment Based Precision | 0,72039 | 0,76798 |
| Best Align | 0,66892 | 0,87443 |
| Behavioral Generalization | 0,73603 | 0,91608 |
| Behavioral Precision | 0,75170 | 0,76383 |
| Behavioral Recall | 0,89596 | 0,68228 |
| Token Based Fitness | 0,87430 | 0,69600 |
| Alignment Based Generalization | 0,00829 | 0,05993 |
| One Align | 0,99284 | 0,78688 |

| | | |
|---|---|---|
| **Totale Communaliteit** | 0,7056628 | **KMO** |
| **Bartlett's p-waarde** | 0,00000 | 0,7608805 |
| **RMSR** | 0,03879 | |

*Tabel 6: Samenvatting kwaliteit van factoranalyse experiment 1*

Om de algemene kwaliteit en zinvolheid van de factoranalyse te beoordelen wordt er voornamelijk gebruik gemaakt van enerzijds de KMO en anderzijds de MSA-waarden per gemeten variabele.

De Kaiser-Meyer-Olkin (KMO) statistiek [38] levert een index, tussen 0 en 1, die de proportie van variantie weergeeft tussen de variabelen. Deze statistiek gaat testen of de partiele correlaties tussen de items klein zijn. Als twee variabelen een gedeelde factor hebben met andere variabelen dan zijn hun partiële correlatie laag, wat impliceert dat ze een unieke variantie delen met elkaar. De index zal 1 zijn indien elke variabele perfect voorspeld kan worden zonder fout door de andere variabelen. De interpretatie van deze statistiek wordt weergegeven in volgende tabel:

| **KMO Value** | **Degree of Common** |
|---|---|
| 0,90 tot 1,00 | Marvelous |
| 0,80 tot 0,89 | Meritorious |
| 0,70 tot 0,79 | Middling |
| 0,60 tot 0,69 | Mediocre |
| 0,50 tot 0,59 | Miserable |
| 0,00 tot 0,49 | Don't Factor |

*Tabel 7: Interpretatie KMO-statistiek*

Voor dit experiment bedraagt de KMO 0,76 wat gemiddeld is. Het is dus, volgens deze maatstaf, zinvol om met de huidige data een factoranalyse uit te voeren.

De Measures of Sampling Adequacy (MSA) [38] meten als het ware hetzelfde als de KMO, maar dan per variabele die in de factoranalyse wordt opgenomen. Deze waarden zullen verhogen indien (1) de sample grootte verhoogt, (2) de gemiddelde correlaties verhogen, (3) het aantal variabelen verhoogt of (4) het aantal factoren verhoogt. De MSA per variabele zou minstens 0,5 moeten bedragen. De MSA-waarden voor dit experiment zijn voor alle metrieken voldoende hoog, met uitzondering van de Alignment Based Generalization. In principe zou deze metriek uit de factoranalyse verwijderd moeten worden, maar aangezien de KMO-waarde voldoende hoog is, is dit niet vereist [38].

Een volgende stap is om per uitgevoerde factoranalyse verschillende statistieken te berekenen, die opnieuw de kwaliteit en/of betrouwbaarheid van de gevonden factoren gaan weergeven. Zo kan bijvoorbeeld gekeken worden naar de Root Mean Squared Residual (RMSR) [41]. Deze statistiek geeft een schatting van de gemiddelde mate van het verschil tussen de corresponderende elementen in de factoranalyse. Deze waarde zou kleiner moeten zijn dan 0,06, wat voor deze factoranalyse het geval is.

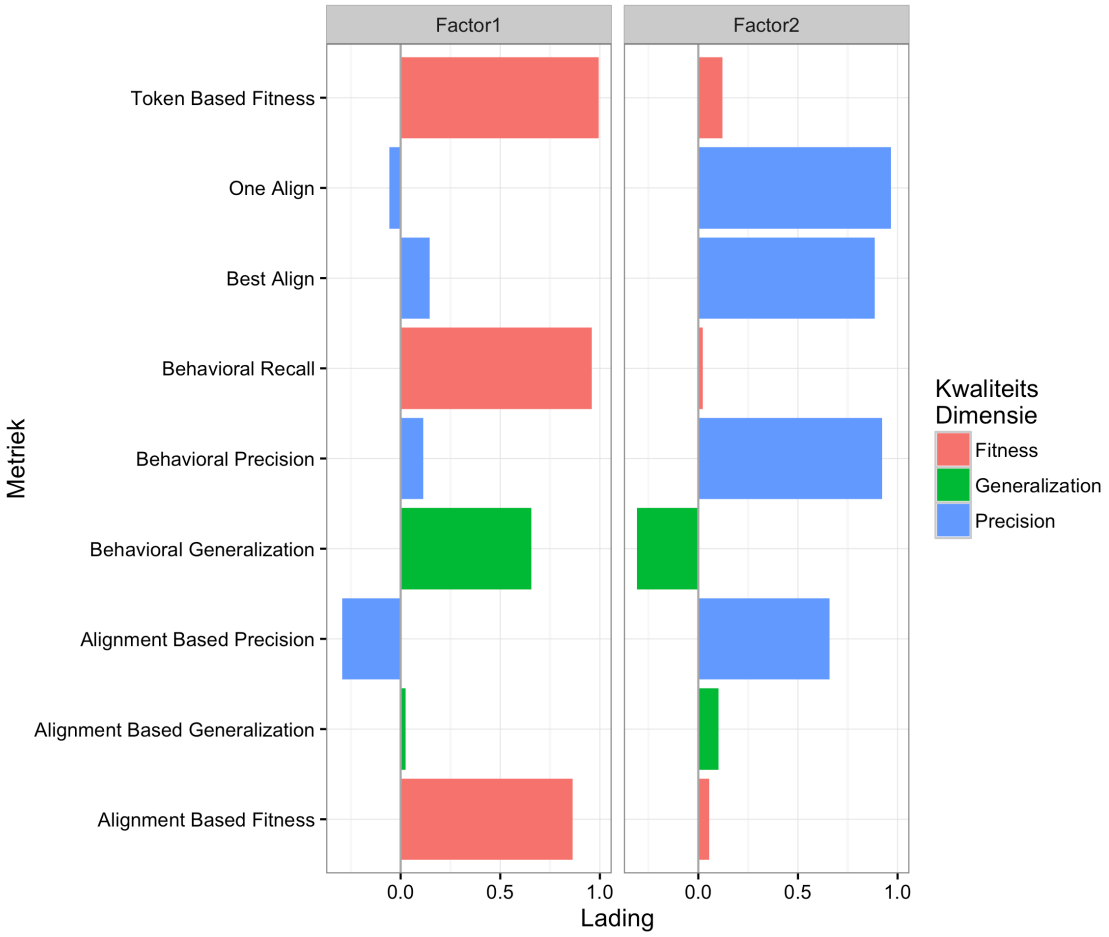
Vervolgens wordt vaak gekeken naar de Bartlett's test of Sphericity [38], [42]. Deze test gaat nagaan in welke mate de correlatiematrix van de data, overeenkomt met een eenheidsmatrix. Als dit het geval zou zijn, dan heeft het uiteraard geen zin om factoren te gaan opstellen omdat er dan evenveel factoren als variabelen gevonden zullen worden. Voor dit experiment heeft de Bartlett's test een p-waarde van 0,00 waardoor kan besloten worden dat we niet te maken hebben met een eenheidsmatrix en het bijgevolg zinvol is om een factoranalyse uit te voeren.

Tot slot moet gekeken worden naar de hoeveelheid variantie die verklaard wordt door het vormen van factoren, zowel per variabele als in het totaal. Dit wordt weergegeven aan de hand van de communaliteiten [38], [39], [41], [43]. Voor de uitgevoerde analyse bedraagt de totale communaliteit ongeveer 0,70, of met andere woorden 70% van de variantie in de variabelen wordt verklaard door gebruik te maken van twee factoren. Als gekeken wordt naar de individuele communaliteiten per metriek blijkt dat bij experiment 1 slechts één metriek is waarvan minder dan 50% van de variantie verklaard worden, namelijk bij Alignment Based Generalization. De variantie in deze metriek die verklaard wordt door het vormen van de factoren is zelfs verwaarloosbaar.

Merk eveneens op dat voor deze factoranalyses gekozen werd om de Promax rotatie toe te passen. Dit is namelijk een oblique rotatiemethode die uitgaat van eventuele correlatie tussen de factoren. Er wordt gekozen voor deze methode omdat het gaat om een exploratieve factoranalyse en er dus nog niet geweten is of er wel degelijk correlatie is tussen de verschillende factoren. Als er geen correlatie zou zijn, dan is de keuze voor deze methode nog steeds verantwoord. Als er echter wel correlatie zou zijn tussen de factoren en er toch een orthogonale rotatiemethode zou toegepast worden, kan dit een vertekend beeld geven over de analyse. [38]

#### 4.1.3.3 Factoranalyse

De ladingen van de factoranalyse voor experiment kunnen als volgt geplot worden:




*Figuur 21: Twee-factoranalyse experiment 1*

De twee gevonden factoren reflecteren duidelijk de kwaliteitsdimensies fitness en precision. Op factor 1 laden Token Based Fitness, Behavioral Recall en Alignment Based Fitness zeer hoog, maar ook Behavioral Generalization heeft een relatief hoge lading[6] op deze factor. Ze hebben elk een lading van minstens 0,6[7]. Er kan dus net zoals in de correlatieanalyse gesteld worden dat deze laatste metriek niet de dimensie generalization meet, zoals de theorie beschrijft, maar eerder de dimensie fitness. Merk eveneens op dat deze metriek negatief gecorreleerd is met factor 2, een factor waarop One Align, Behavioral, Best Align en Alignment Based Precision sterk laden. De dimensie generalization, die volgens de theorie gemeten wordt door Behavioral Generalization en Alignment Based Generalization is niet terug te vinden in deze factoranalyse. De Alignment Based Generalization laadt zelfs op geen van de gevonden factor sterk.

[6] Factor ladingen zijn zeer gelijkaardig aan de gewichten in multiple regressie analyse, ze stellen de kracht van de correlatie voor tussen de variabele en de factor. [43]

[7] Een lading van 0,3 is bij de gebruikte samplegrootte reeds statistisch significant volgens Hair [38]. Omwille van praktische significantie, is echter een factor lading van ongeveer 0,5 aangeraden.

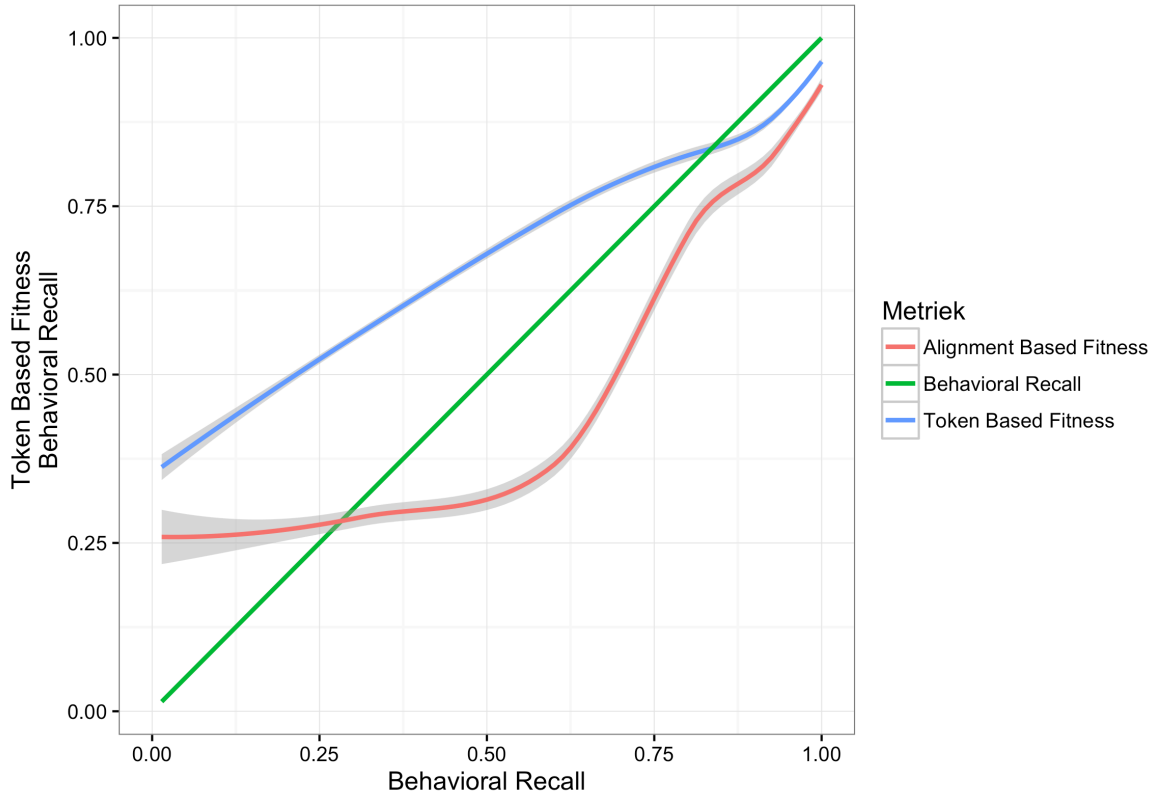
De correlaties tussen de twee factoren worden samengevat in de volgende tabel. Er valt een duidelijke negatieve correlatie op tussen de gevonden factoren. Aangezien de factoren geïnterpreteerd zouden kunnen worden als de dimensies fitness en precision zou deze negatieve correlatie tussen de factoren kunnen wijzen op een negatieve relatie tussen deze dimensies.

| | Factor 1 | Factor 2 |
|---|---|---|
| Factor 1 | 1,0000 | -0,523 |
| Factor 2 | -0,523 | 1,0000 |

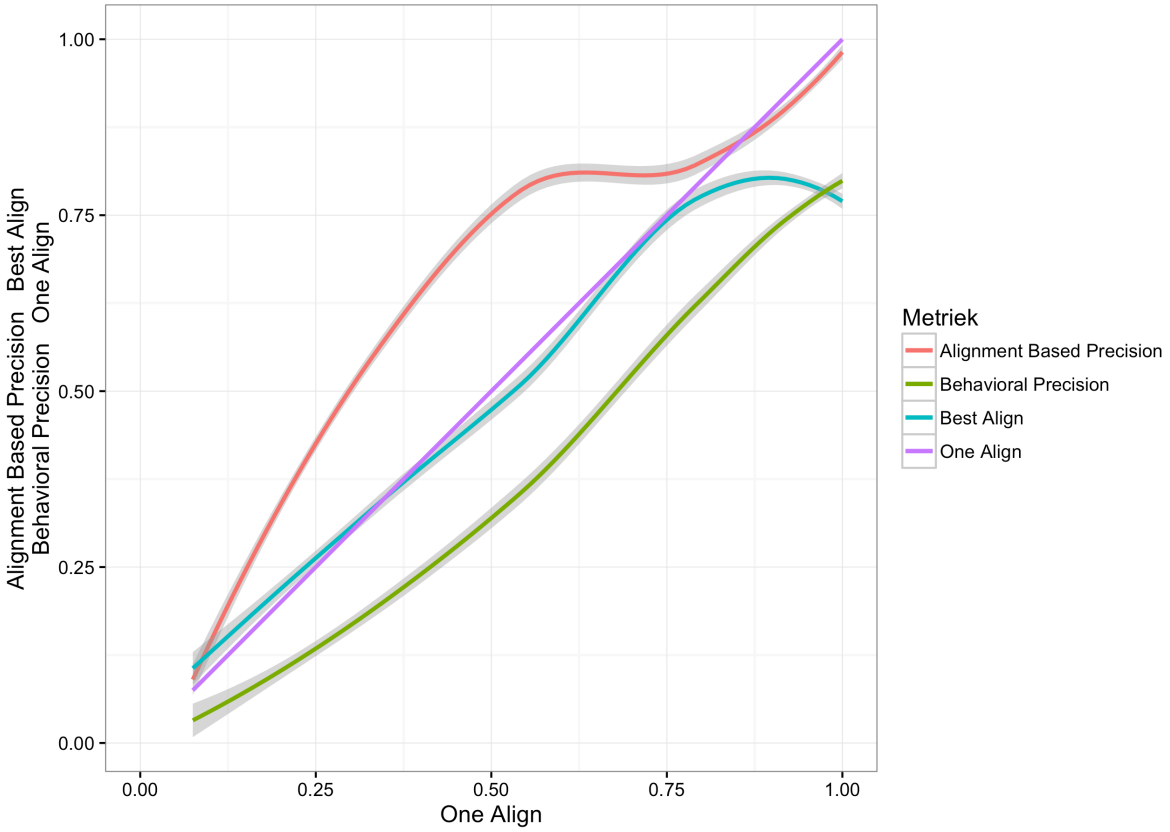
*Tabel 8: Correlaties tussen de factoren (experiment 1)*

## 4.2 Gevoeligheid aan het niveau van kwaliteit

In deze sectie wordt onderzocht hoe elk van de metrieken reageert op een bepaald niveau van fitness of precision. Om de gevoeligheid aan de kwaliteit van het model van elk van de geobserveerde metrieken te meten ten opzichte van hun respectievelijke dimensie, zouden de metrieken geplot kunnen worden ten opzichte van de gebruikte verhouding in het Venn diagram in sectie 2.1 voor hun respectievelijke dimensie. Nochtans is dit niet mogelijk aangezien deze formule niet expliciet berekend kan worden. Daarom worden de metrieken binnen een dimensie relatief ten opzichte van elkaar weergegeven. Op deze manier kan gekeken worden hoe elk van de metrieken reageren ter hoogte van een bepaald niveau van fitness of precision.




*Figuur 22: Gevoeligheid aan kwaliteit van het model voor fitness*




*Figuur 23: Gevoeligheid aan kwaliteit van het model voor precision*

Bovenstaande figuren geven de waarden van de metrieken weer, gegeven een bepaalde referentiemetriek. De gebruikte referentiemetrieken zijn in dit geval Behavioral Recall en One Align Precision. Deze referentiemetrieken werden gekozen op een willekeurige basis.

Uit Figuur 22 kan worden opgemaakt dat de verschillende fitness-metrieken relatief dicht bij elkaar liggen indien de fitness hoog is (> 0,85). Van zodra de fitness daalt, liggen de metrieken verder weg van elkaar. Het patroon van Behavioral Recall en Token Based Fitness is zeer gelijkaardig. Enkel in het gebied tussen 0,75 en 0,9 vertoond deze laatste metriek een afvlakkend patroon. Als de Alignment Based Fitness beschouwd wordt, blijkt eveneens dat deze in bepaalde gebieden een sterk afvlakkend patroon vertoond, relatief ten opzichte van de Behavioral Recall. In het gebied tussen 0 en ongeveer 0,5 stijgt deze metriek slechts gering, terwijl zowel de Behavioral Recall als de Token Based Fitness sterk toenemen. In het gebied tussen 0,5 en 0,8 neemt de Alignment Based Fitness echter veel sterker toe dan de andere twee metrieken. In het algemeen kwantificeert de Token Based Fitness deze dimensie hoger dan de andere twee metrieken, terwijl de Alignment Based Fitness in het grootste gedeelte de fitness lager kwantificeert.

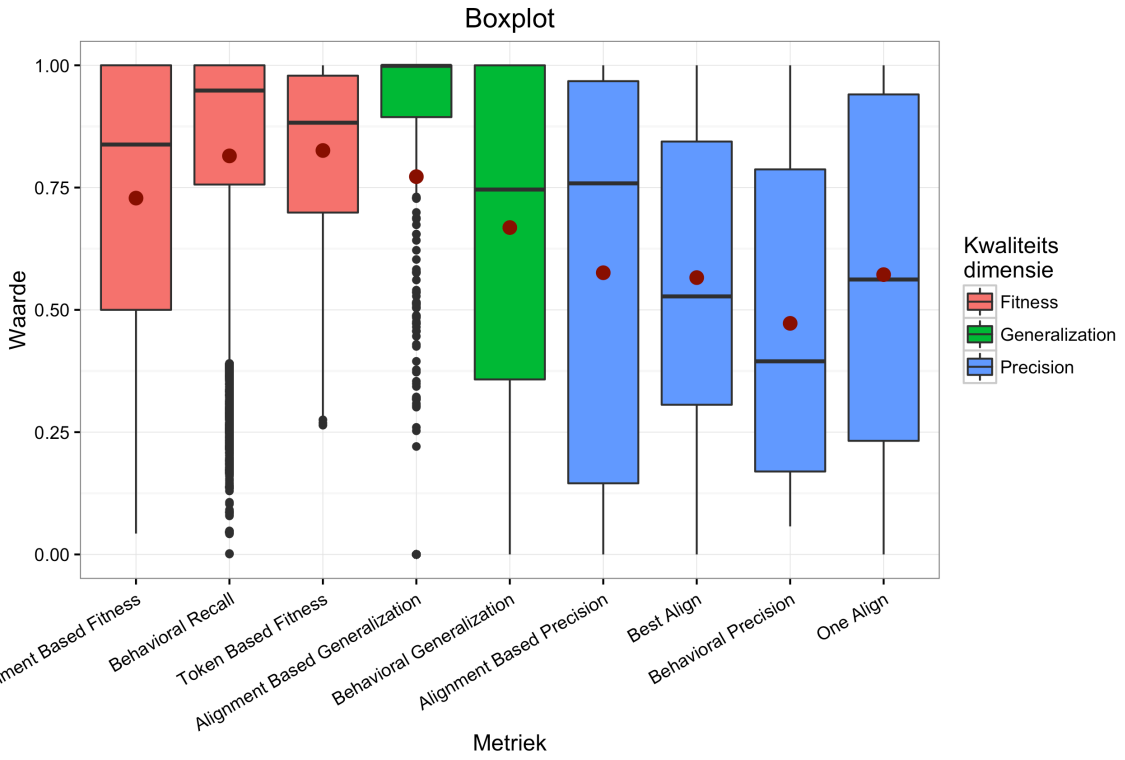
Uit Figuur 23 wordt duidelijk dat One Align en Best Align Precision relatief dicht bij elkaar liggen over het gehele gebied. Nochtans wijken ze van elkaar bij een precision hoger dan ongeveer 0,85. In geval van zulk een hoge precision liggen One Align en Alignment Based Precision zeer kort bij elkaar, maar ook Best Align en Behavioral Precision liggen in dit gebied relatief kort bij elkaar. In het algemeen valt op dat Alignment Based Precision gewoonlijk de precision hoger kwantificeert dan de andere precision-metrieken. De Behavioral Precision daarentegen, kwantificeert precision over het

gehele gebied lager dan de andere precision-metrieken. Verder valt op dat Best Align Precision in geval van een zeer goed model, en dus hoge precision, afwijkt van de One Align Precision en plots de precision lager gaat kwantificeren. De reden hiervoor is echter onduidelijk. De hellingen van elk van deze metrieken zijn in het gebied tussen 0 en ongeveer 0,5 zeer gelijkaardig, zonder afwijkende patronen. Nochtans vertoond de Alignment Based Precision een sterk afvlakkend patroon in het gebied tussen 0,5 en 0,75. Ook de Best Align vertoond een gelijkaardig gedrag in het gebied tussen 0,75 en 1. In het allerlaatste gedeelte van dit gebied, daalt deze metriek zelfs. Dit betekent dat deze metriek de precision plotseling lager gaat kwantificeren, terwijl de andere precision-metrieken toch een positief patroon vertonen.

Uit bovenstaande analyse blijkt dat de metrieken zich anders gedragen bij een model met hoge kwaliteit dan bij een model met een relatief lage kwaliteit. Daarom worden bovenstaande analyses opnieuw uitgevoerd, dit keer voor de twee datasets van experiment 1 apart. Er wordt dus opnieuw een correlatie- en factoranalyse uitgevoerd voor enerzijds een dataset met relatief eenvoudige modellen (experiment 1a) en anderzijds voor een dataset met meer complexe modellen (experiment 1b). Op deze manier kan gekeken worden of er andere resultaten gevonden worden voor beide datasets van experiment 1.

## 4.3 Analyse experiment 1a

### 4.3.1 Samenvattende analyse

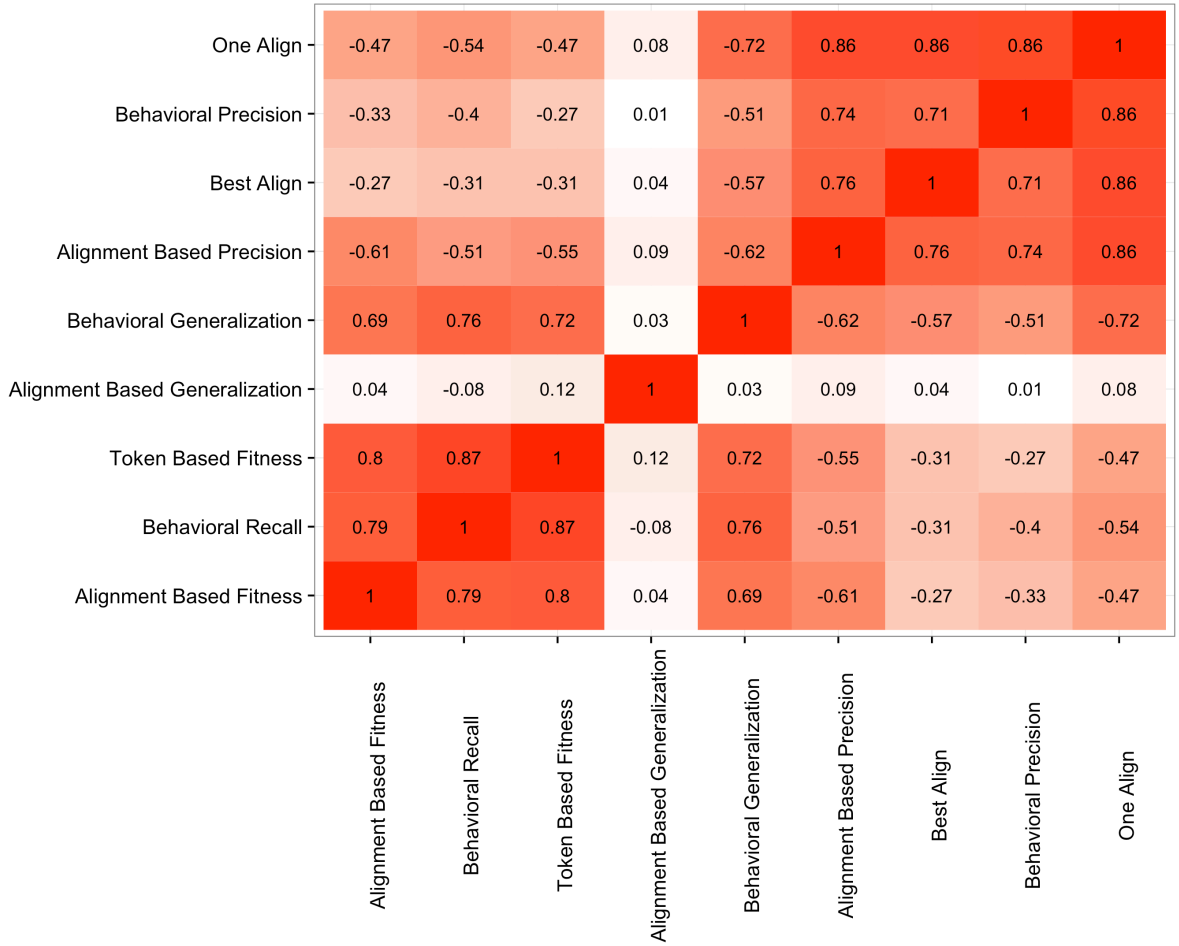



*Figuur 24: Boxplot van data van experiment 1a*

Bovenstaande figuur geeft de data weer van experiment 1a. Het wordt duidelijk dat de data van experiment 1b, als er gekeken wordt naar de metrieken binnen één bepaalde dimensie. Zo liggen

bijvoorbeeld de drie fitness-metrieken relatief dicht bij elkaar, maar ook de gemiddelden van de metrieken die de precision-dimensie meten, liggen kort bij elkaar. Ook de spreiding binnen een bepaalde dimensie is relatief consistent, enkel bij de dimensie generalization is er een groot verschil in spreiding tussen de twee metrieken.

### 4.3.2 Correlatieanalyse




*Figuur 25: Correlatiematrix voor experiment 1a*

Uit bovenstaande correlatiematrix kunnen twee duidelijke groepen worden opgemaakt van sterk correlerende metrieken. De eerste is een groep van metrieken waarvan gesteld kan worden dat deze de fitness-dimensie meten. In deze groep zitten de Token Based en Alignment Based Fitness, Behavioral Recall en de Behavioral Generalization. Ook de Behavioral Generalization kan worden toegevoegd aan deze groep van fitness-metrieken.

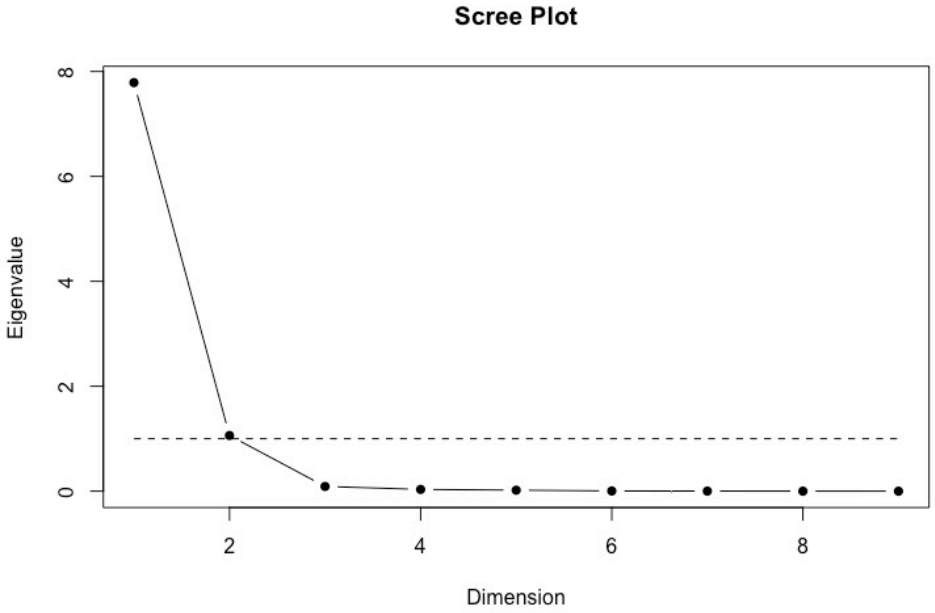
Van de tweede groep van sterk gecorreleerde metrieken kan gesteld worden dat deze de precision-dimensie meten. In deze groep zitten de One Align, Best Align, Behavioral en Alignment Based Precision. Wat eveneens opvalt is dat de Behavioral Generalization relatief sterk negatief gecorreleerd is met deze groep. Dit is een resultaat dat eveneens in experiment 1 gevonden werd.

Vervolgens blijkt dat de twee groepen van metrieken die zonet besproken werden, negatief gecorreleerd zijn aan elkaar. Met andere woorden, volgens deze correlatiematrix zijn de dimensie fitness en precision negatief gerelateerd aan elkaar. Ook dit resultaat kon worden opgemaakt uit de correlatiematrix van experiment 1.

Tot slot is het zeer duidelijk dat Alignment Based Generalization, net zoals bij experiment 1, niet gecorreleerd is aan eender welke andere geobserveerde metriek. Opnieuw een aanwijzing dat deze metriek een andere dimensie meet die de andere geobserveerde metrieken niet meten. Uit de literatuur kon ook worden opgemaakt dat de dimensies generalization en precision negatief gecorreleerd zijn aan elkaar. Dit is echter enkel het geval voor de metriek Behavioral Generalization en dus niet voor de Alignment Based Generalization.

### 4.3.3 Factoranalyse

Om een factoranalyse uit te voeren van deze data, zal dezelfde procedure als die van experiment 1 gevolgd worden. Er zal dus eerst bepaald worden hoeveel factoren gebruikt zullen worden aan de hand van een scree plot. Vervolgens wordt de kwaliteit van de analyses beoordeeld om tot slot te eindigen met het in detail bespreken van de gekozen factoranalyse.




*Figuur 26: Scree plot experiment 1a*

In de scree plot valt een duidelijke knik op net voor factor 3. Er zou dus bij dit experiment gebruik gemaakt kunnen worden van drie factoren. Op basis van het criterium dat de eigenwaarde hoger moet zijn dan één, zou gekozen worden voor twee factoren. Rekening houdende met de interpreteerbaarheid en de kwaliteit van de factoranalyse, wordt uiteindelijk gekozen voor een twee-factoranalyse.

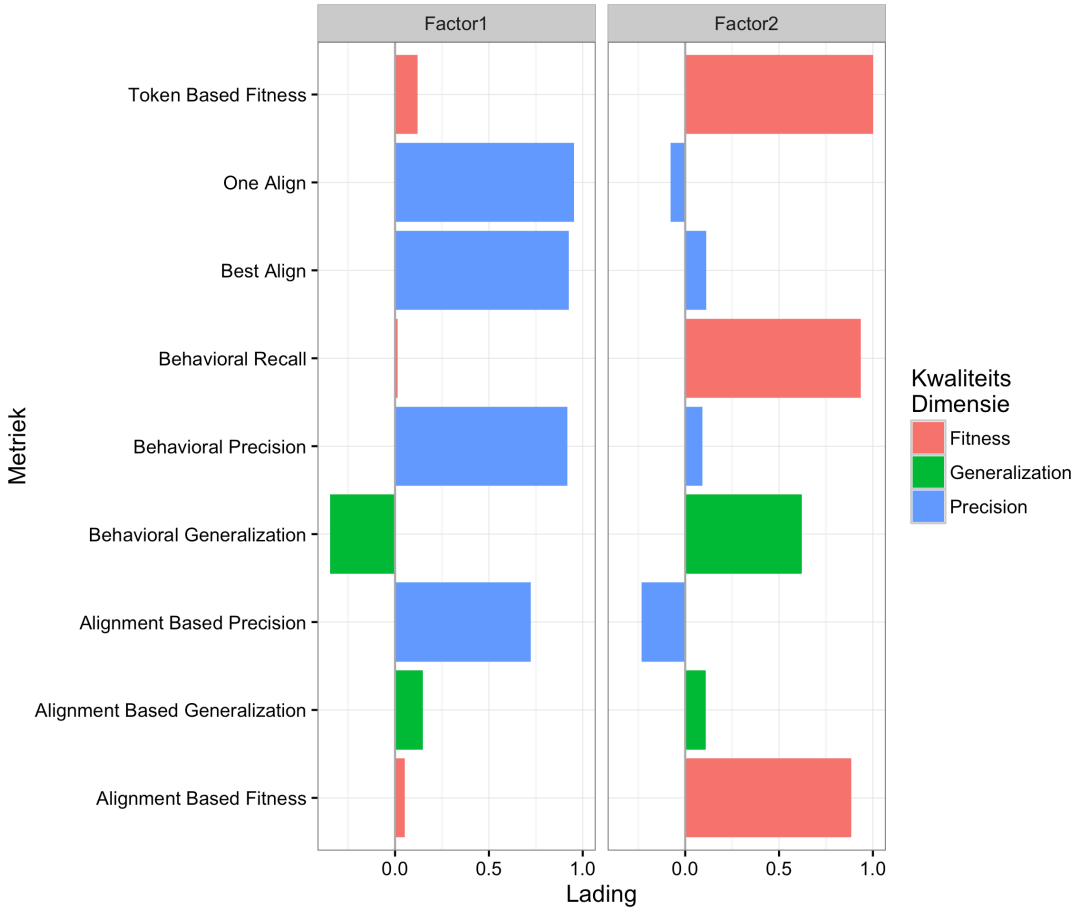
| | Communaliteit | |
|---|---|---|
| **Metriek** | 2 Factoren | **MSA** |
| Alignment Based Fitness | 0,74024 | 0,82291 |
| Alignment Based Generalization | 0,01681 | 0,05015 |
| Alignment Based Precision | 0,76074 | 0,76665 |
| Best Align | 0,75669 | 0,85673 |
| Behavioral Generalization | 0,73926 | 0,88510 |
| Behavioral Precision | 0,76113 | 0,73878 |
| Behavioral Recall | 0,85507 | 0,68383 |
| One Align | 0,99503 | 0,80079 |
| Token Based Fitness | 0,88967 | 0,70331 |

| | | |
|---|---|---|
| **Totale Communaliteit** | 0,72384 | **KMO** |
| **Bartlett's p-waarde** | 0,00000 | 0,75838 |
| **RMSR** | 0,04037 | |

*Tabel 9: Samenvatting kwaliteit twee-factoranalyse (experiment 1a)*

Net zoals in het vorige experiment is ook hier de KMO voldoende hoog om een zinvolle factoranalyse uit te voeren. In totaal wordt ongeveer 72% van de totale variantie verklaard door het vormen van de twee factoren. Enkel de variantie in de Alignment Based Generalization wordt nauwelijks verklaard, wat te verwachten was op basis van de correlatieanalyse. Ten slotte zijn zowel de Bartlett's test en de RMSR beiden laag genoeg, wat erop wijst dat het zinvol is om deze factoranalyse uit te voeren.

Ook bij dit experiment zullen de ladingen van de metrieken op de factoren op eenzelfde manier geplot worden. Merk eveneens op dat ook hier gebruik gemaakt werd van de Promax rotatie, omwille van dezelfde reden als in experiment 1.




*Figuur 27: Twee-factoranalyse experiment 1a*

De twee gevonden factoren reflecteren zeer duidelijk de precision en de fitness-dimensie. Op factor 1 laden alle vier de geobserveerde precision-metrieken zeer hoog. Wat tenslotte ook opvalt in deze factor is dat Behavioral Generalization negatief laadt op deze factor, ook al is de lading relatief laag (≈ 0,3). Dit zou een bevestiging kunnen zijn van de relatie tussen precision en generalization die gevonden werd in de literatuurstudie.

Op factor 2 laden Token Based Fitness, Behavioral Recall en Alignment Based Fitness zeer sterk, maar ook Behavioral Generalization laadt sterk op deze factor. Dit is een aanwijzing dat Behavioral Generalization helemaal niet de generalization-dimensie meet, maar eerder de fitness-dimensie. Een andere mogelijke verklaring voor dit patroon kan uitgelegd worden met behulp van het Venn diagram in sectie 2.1. Als het systeem ongeveer gelijk zou zijn aan het model (in het geval van weinig noise en redelijk complete logs), dan is de fitness ongeveer gelijk aan de generalization. Nochtans is er in de meeste logs wel een aanzienlijke hoeveelheid noise aanwezig en zijn deze logs vaak incomplete waardoor het systeem en het model minder goed met elkaar overeenkomen. Aangezien in dit experiment meer eenvoudige modellen gebruikt worden met relatief weinig noise en redelijk complete logs, zullen fitness en generalization meer gelijk zijn aan elkaar waardoor ze zich in dezelfde factor bevinden.

In het algemeen kan uit deze analyse besloten worden dat bij een goede kwaliteit van het model, de geobserveerde metrieken die de kwaliteitsdimensies fitness en precision meten wel degelijk hun

respectievelijke dimensie meten. Dit resultaat is een empirische bevestiging van de literatuur. De metrieken die de dimensie generalization kwantificeren daarentegen, meten iets anders dan wat ze volgens de literatuur beweren te meten. Voornamelijk Behavioral Generalization lijkt eerder de dimensie fitness te meten dan generalization. Over de Alignment Based Generalization is het moeilijker om een conclusie te trekken aangezien deze zich anders gedraagt dan de andere geobserveerde metrieken.

In onderstaande tabel worden, net zoals in experiment 1, de correlaties tussen de factoren weergegeven. De gevonden correlatie tussen de twee factoren is sterk negatief, wat zou kunnen wijzen op een negatieve relatie tussen fitness en precision.

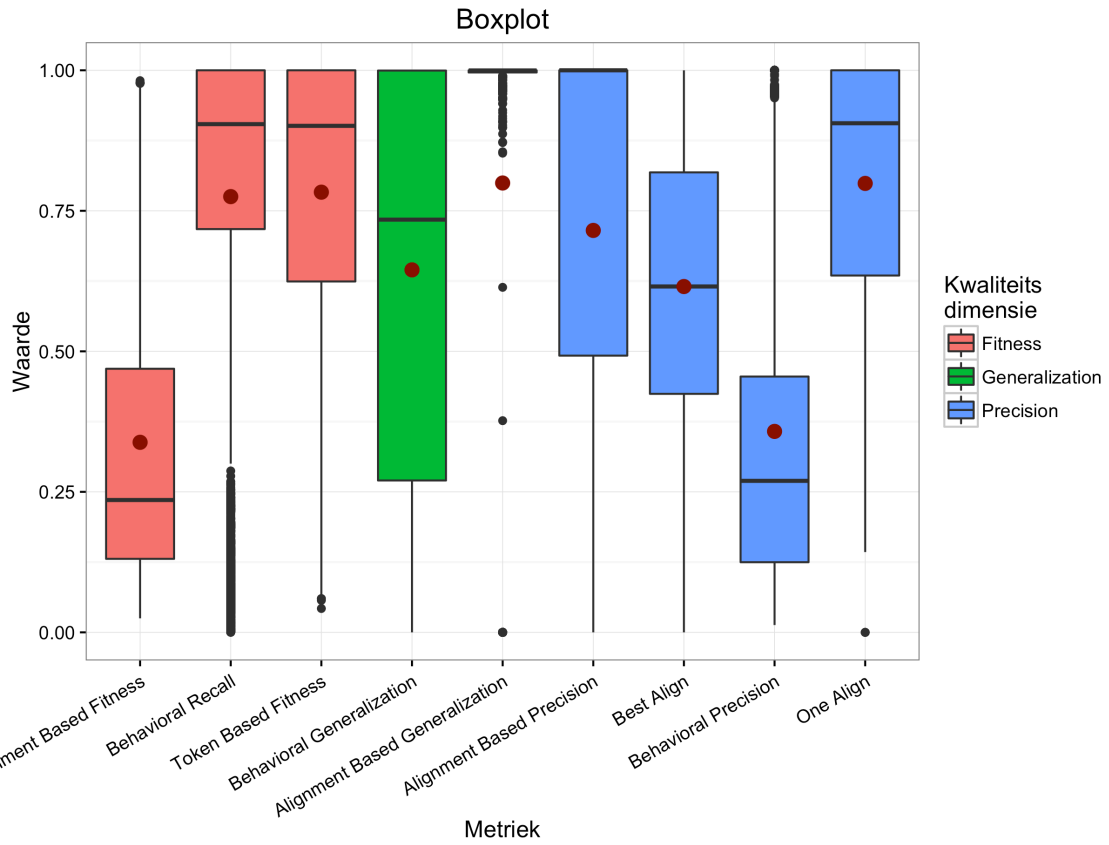
| | Factor 1 | Factor 2 |
|---|---|---|
| Factor 1 | 1,0000 | -0,5410 |
| Factor 2 | -0,5410 | 1,0000 |

*Tabel 10: Correlaties tussen de factoren (experiment 1a)*

Op basis van bovenstaande correlaties tussen de twee factoren wordt duidelijk dat de keuze om een oblique rotatiemethode toe te passen weldegelijk verantwoord was. Doordat deze substantiële correlatie aanwezig is, zou het gebruik van een orthogonale rotatiemethode niet toegestaan zijn.

## 4.4 Analyse experiment 1b

### 4.4.1 Samenvattende analyse




*Figuur 28: Boxplot van data van experiment 1b*

Uit bovenstaand boxplot van experiment 1a wordt duidelijk dat de bekomen waarden voor de metrieken, die een bepaalde kwaliteitsdimensie meten, niet consistent zijn. Voor bijvoorbeeld precision zitten er grote verschillen in de gemiddelden van de metingen. Zo is het gemiddelde van Behavioral Precision veel lager dan de andere gemiddelden van de metrieken voor deze kwaliteitsdimensie. De metriek Behavioral Precision heeft een gemiddelde van ongeveer 0,35, terwijl de andere precision-metrieken hier ver boven liggen. Hetzelfde patroon geldt eveneens voor de dimensie fitness. Bijvoorbeeld de Alignment Based Fitness kwantificeert de dimensie fitness gemiddeld gezien lager dan de andere fitness-metrieken. De Behavioral Recall en Token Based Fitness hebben daarentegen een redelijk overeenkomende boxplot.

Voor de dimensie generalization is de spreiding binnen de Behavioral Generalization veel groter dan bij de Alignment Based Generalization. Dit resultaat werd eveneens gevonden in de eerder opgestelde boxplots van experiment 1 en 1a.

In het algemeen wordt duidelijk dat de metrieken binnen één dimensie bij modellen van lagere kwaliteit verder uit elkaar liggen dan oorspronkelijk verwacht uit Figuur 18. Hier lagen de gemiddelden van de metrieken binnen één dimensie namelijk veel korter bij elkaar. Verder valt eveneens op dat de dimensies fitness en generalization hoger gekwantificeerd worden in experiment 1a dan in experiment 1b. Dit kan het gevolg zijn van het feit dat de gekozen systemen voor dit

tweede deel van experiment 1 bewust complexer werden gehouden, waardoor de algoritmes meer moeite hebben om een degelijk model te ontdekken uit de event logs.

### 4.4.2 Correlatieanalyse


*Figuur 29: Correlatiematrix voor experiment 1b*

Bovenstaande figuur geeft opnieuw de correlatiematrix weer van de verschillende metrieken, ditmaal met de data uit experiment 1b. Hieruit valt duidelijk op dat de verschillende alignment based metrieken, relatief zwak gecorreleerd zijn aan de andere metrieken. Voornamelijk Alignment Based Generalization heeft met geen enkele metriek een correlatie hoger dan 0,14, een resultaat dat ook al eerder werd gevonden.

Verder wordt eveneens duidelijk uit de correlatiematrix dat drie van de vier precision-metrieken die opgenomen werden in de analyse, sterk gecorreleerd zijn aan elkaar. Voornamelijk One Align en Behavioral Precision zijn zeer sterk gecorreleerd aan elkaar, maar ook Best Align Precision kan aan deze groep toegevoegd worden. Zoals eerder vermeld is de Alignment Based Precision nauwelijks gecorreleerd aan de andere precision-metrieken en valt dus buiten deze groep. Nochtans is deze metriek relatief sterk gecorreleerd aan One Align Precision. In de correlatieanalyse van experiment 1 en 1a was deze metriek nochtans wel sterk gecorreleerd met de andere precision-metrieken. Een

laatste bemerking voor deze metriek is dat deze relatief sterk negatief gecorreleerd is met de Behavioral Recall.

Er valt vervolgens een tweede groep van sterk gecorreleerde metrieken op, namelijk Behavioral Recall, Token Based Fitness en Behavioral Generalization. Deze laatste metriek zou volgens de theorie echter een andere dimensie moeten meten dan de andere twee metrieken in deze groep. Dit is opnieuw een aanwijzing dat Behavioral Generalization iets anders zou meten dan verwacht werd uit de literatuurstudie. De Alignment Based Fitness zou in principe ook kunnen toegevoegd worden aan deze groep, ook al is deze metriek slechts relatief licht gecorreleerd met Behavioral Recall en Token Based Fitness. Er wordt dus een gelijkaardig resultaat gevonden als bij de correlatieanalyse van experiment 1 en 1a.

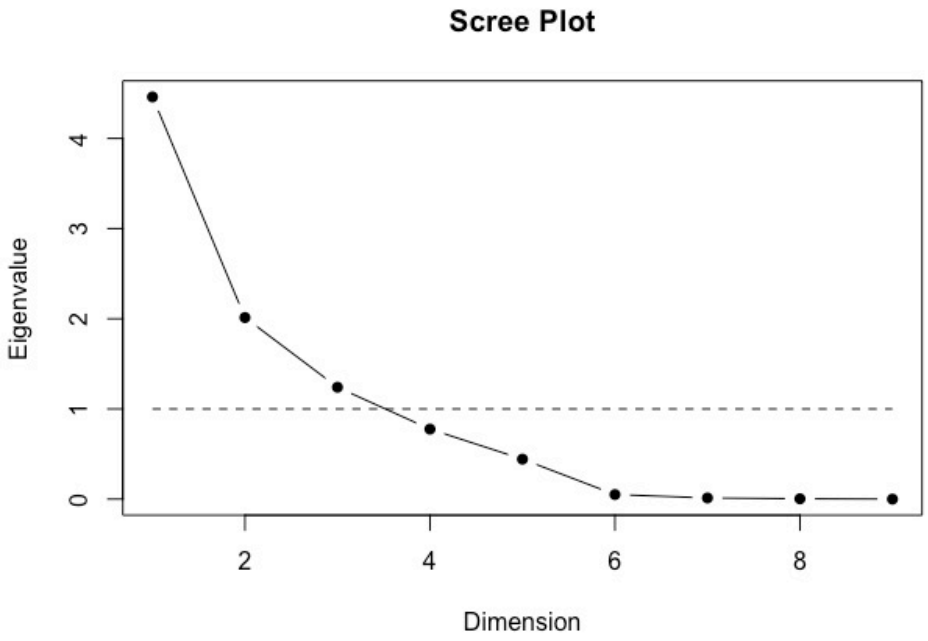
Ook in deze analyse is het moeilijk om de derde dimensie generalization op te maken. De twee metrieken die deze dimensie volgens de literatuur zouden moeten meten, zijn ofwel gecorreleerd met een andere dimensie (Behavioral Generalization), ofwel helemaal niet gecorreleerd met de andere metrieken (Alignment Based Generalization).

Verder zou uit de theorie verwacht kunnen worden dat de metrieken die generalization en precision meten, negatief aan elkaar gecorreleerd zouden zijn. Uit bovenstaande correlatiematrix blijkt dit echter niet zo te zijn. Het blijkt daarentegen dat fitness en generalization[8] positief gecorreleerd zijn aan elkaar. Dit is een bevinding die niet terug te vinden is in de literatuur. Er zijn ook enkele precision-metrieken die negatief gecorreleerd zijn aan enkele fitness-metrieken, namelijk One Align en Alignment Based Precision zijn negatief gecorreleerd aan Behavioral Recall en Token Based Fitness.

### 4.4.3 Factoranalyse

Er wordt in deze sectie opnieuw een factoranalyse uitgevoerd op dezelfde manier en werkwijze als bij experiment 1. Er zal opnieuw aan de hand van een scree plot bepaald worden van hoeveel factoren gebruik gemaakt zal worden, rekening houdende met de interpreteerbaarheid van de factoren. Vervolgens zal er opnieuw een analyse van de kwaliteit uitgevoerd worden om tot slot te eindigen met het bespreken van de ladingen van de metrieken op de gevonden factoren.

[8] Rekening houdende met Behavioral Generalization en niet met Alignment Based Generalization




*Figuur 30: Scree plot experiment 1b*

Voor experiment 1b is het op basis van bovenstaande scree plot moeilijk om een duidelijke knik te definiëren, maar er zou gezegd kunnen worden dat deze zich op de zesde factor bevindt. Omwille van dit criterium, zou dus gekozen kunnen worden om zes factoren te gebruiken. Op basis van het tweede criterium, namelijk op dat de eigenwaarde minimum 1 moet bedragen, zou gekozen worden voor drie factoren. Er wordt uiteindelijk gekozen om een twee-, drie-, vier- en vijf-factoranalyse uit te voeren, omdat naast de twee bovengenoemde criteria ook rekening gehouden moet worden met de interpreteerbaarheid van de factoren. Daarom wordt gekozen om de verschillende factoranalyses op te stellen die eveneens in bijlage gevonden kunnen worden. Uiteindelijk wordt uit deze analyses gekozen om de vier-factoranalyse verder in detail te bespreken.

Net zoals in de analyse van experiment 1 en 1a, wordt ook de kwaliteit van de analyse gecontroleerd die samengevat wordt in de volgende tabel:

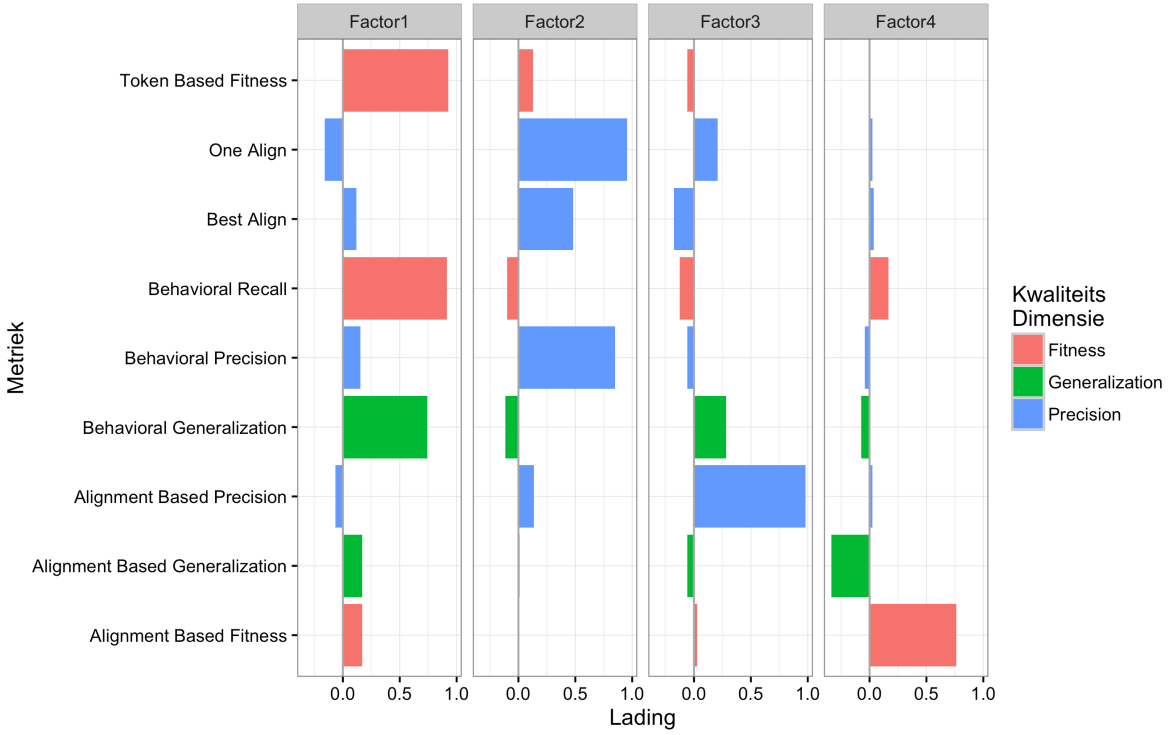
| | **Communaliteit** | |
|---|---|---|
| **Metriek** | 4 Factoren | **MSA** |
| Alignment Based Fitness | 0,63569 | 0,48129 |
| Alignment Based Generalization | 0,12181 | 0,50387 |
| Alignment Based Precision | 0,97327 | 0,45884 |
| Best Align | 0,28920 | 0,51203 |
| Behavioral Generalization | 0,56737 | 0,64161 |
| Behavioral Precision | 0,75313 | 0,54132 |
| Behavioral Recall | 0,97919 | 0,56826 |
| One Align | 0,99501 | 0,42527 |
| Token Based Fitness | 0,90503 | 0,57598 |

| | | |
|---|---|---|
| **Totale Communaliteit** | 0,69108 | **KMO** |
| **Bartlett's p-waarde** | 0,00000 | 0,52910 |
| **RMSR** | 0,02803 | |

*Tabel 11: Samenvatting kwaliteit vier-factoranalyse (experiment 1b)*

De KMO-waarde wordt geïnterpreteerd als miserabel, maar is wel voldoende hoog om een zinvolle factoranalyse uit te voeren. Als gekeken wordt naar de individuele MSA-waarden van de metrieken zijn de meeste voldoende hoog, met uitzondering van Alignment Based Fitness, Alignment Based Precision en One Align Precision. In totaal wordt ongeveer 69% van de totale variantie verklaard door het vormen van vier factoren. Ook de individuele communaliteiten zijn voldoende hoog voor elk van de metrieken met uitzondering van de Alignment Based Generalization en Best Align Precision. Ten slotte zijn zowel de Bartlett's test en de RMSR beiden laag genoeg. In het algemeen kan besloten worden dat dit een zinvolle factoranalyse.

Ook bij deze factoranalyses werd gebruik gemaakt van de Promax rotatie, omwille van dezelfde reden als in de factoranalyse van experiment 1. Na het uitvoeren van deze vier-factoranalyse, kunnen de factor ladingen geplot worden zoals in de volgende figuur:




*Figuur 31: Vier-factoranalyse experiment 1b*

Het wordt op basis van deze factoranalyse duidelijk dat factor 1 eerder de fitness-dimensie voorstelt. Zowel Token Based Fitness als Behavioral Recall hebben een hoge lading op deze factor. Behavioral Generalization heeft echter ook een hoge lading op deze factor. Rekening houdend met de literatuurstudie, is dit een resultaat dat niet verwacht werd. Het is bijgevolg opnieuw een bevestiging dat deze metriek niet de dimensie generalization meet, maar eerder de dimensie fitness. De laatste geobserveerde fitnessmetriek, Alignment Based Fitness, laadt echter hoog op factor 4 en meet dus iets anders dan de dimensie fitness in het geval van een model met relatief lage kwaliteit.

Factor 2 blijkt de dimensie precision te meten, omdat zowel One Align als Behavioral Precision hoog laden op deze factor. Ook Best Align Precision laadt relatief sterk op deze factor, ook al is het net geen 0,5. De laatste precisionmetriek, Alignment Based Precision, zit (net zoals de Alignment Based Fitness) in een aparte factor, namelijk factor 3. Dit resultaat werd eveneens gevonden in de

correlatieanalyse van dit experiment. Het werd nochtans niet gevonden in de analyse van experiment 1 en 1a, waardoor het aanwijzing zou kunnen dat deze metriek zich anders gedraagt in geval van een model met lagere kwaliteit.

De laatste metriek, die op geen enkele factor sterk laadt, is Alignment Based Generalization. Deze metriek laadt licht negatief op factor 4, maar is niet significant genoeg om conclusies over te trekken. In het algemeen kan gezegd worden dat alle drie de alignment based metrieken andere 'dimensies' meten dan er theoretisch wordt beweerd. De andere metrieken meten volgens deze analyse hun respectievelijke dimensies die ze volgens de literatuur zouden moeten meten, met uitzondering van de Behavioral Generalization. Er zou dus gesteld kunnen worden dat de verschillende alignment based metrieken zich anders gedragen bij modellen met een lagere kwaliteit.

De correlaties tussen de verschillende factoren worden weergegeven in onderstaande tabel. Op basis van deze correlaties kan gesteld worden dat de dimensies fitness en precision (factor 1 en 2) in geval van complexere modellen niet gecorreleerd zijn aan elkaar. Enkel factor 3, waarop enkel de metriek Alignment Based Precision sterk laadt, laadt negatief met de factor 1. Factor 4 ten slotte, waarop de Alignment Based Fitness sterk laadt, heeft een redelijk sterke positieve correlatie met factor 1. Aangezien er op deze twee factoren enkel fitness-metrieken sterk laden, was dit een te verwachten resultaat. Nochtans is het verschil tussen de Alignment Based Fitness enerzijds, en de andere fitness-metrieken anderzijds verschillend genoeg om een aparte factor te vormen.

| | Factor 1 | Factor 2 | Factor 3 | Factor 4 |
|---|---|---|---|---|
| Factor 1 | 1,0000 | 0,0241 | -0,2143 | 0,3794 |
| Factor 2 | 0,0241 | 1,0000 | 0,0180 | 0,0017 |
| Factor 3 | -0,2134 | 0,0180 | 1,0000 | -0,0924 |
| Factor 4 | 0,3794 | 0,0017 | -0,0924 | 1,0000 |

*Tabel 12: Correlaties tussen de factoren (experiment 1b)*

Merk op dat in dit gedeelte van experiment 1 geen sterke negatieve correlatie gevonden werd tussen de factoren die de dimensies fitness en precision weergeven. In dit experiment werd de Alignment Based Precision echter opgenomen in een aparte factor. De negatieve correlatie in experiment 1a zou bijgevolg te wijten kunnen zijn aan deze metriek.

## 4.5 Conclusie

Bovenstaande analyses maken duidelijk dat de metrieken die volgens de theorie de kwaliteitsdimensies fitness en precision meten, zich gelijkaardig gedragen. Token Based Fitness en Behavioral Recall gedragen zich bij zowel bij modellen van hoge als lage kwaliteit zeer gelijkaardig. Alignment Based Fitness ligt in geval van modellen met een hogere kwaliteit ook zeer kort tegen deze twee metrieken. Als de complexiteit echter stijgt en daarbij de kwaliteit van het model, en dus de fitness daalt, gaat deze metriek zich anders gedragen. De Alignment Based Fitness gaat in dit geval de dimensie fitness gemiddeld gezien lager kwantificeren dan de andere fitness-metrieken. Zowel de Token Based Fitness als de Behavioral Recall hebben een zeer gelijkaardig patroon, gaande

van modellen met een lage kwaliteit, tot modellen met een zeer hoge kwaliteit. Beide metrieken zijn dus een betrouwbare maatstaf om de dimensie fitness te kwantificeren. De Alignment Based Fitness daarentegen, vertoonde een zeer afvlakkend effect in geval van modellen met een lage kwaliteit, waardoord deze metriek in die gevallen minder betrouwbaar is. Als de kwaliteit dan stijgt, gaat deze metriek plots meer dan evenredig toenemen.

Bij precision kan een gelijkaardig patroon gevonden worden. Drie van de vier geobserveerde metrieken gedragen zich zeer gelijkaardig, met uitzondering van de Alignment Based Precision. Deze gaat zich in geval van modellen met een lagere kwaliteit, en dus bij een lagere precision, anders gedragen. De Alignment Based Precision gaat in dat geval de precision hoger kwantificeren dan de andere metrieken. De evolutie die de One Align en Behavioral Precision doormaken is sterk gelijkend, terwijl de andere twee metrieken in bepaalde gevallen zich anders gaan gedragen. Zo daalt de Best Align Precision plots in geval van modellen met zeer hoge kwaliteit. Ook de Alignment Based Precision vertoond een afvlakkend effect in geval van modellen met een degelijke kwaliteit. In het algemeen kan geconcludeerd worden dat zowel de One Align als de Behavioral Precision in elk van de gevallen betrouwbare metrieken zijn, terwijl de Best Align en Alignment Based Precision met voorzichtigheid moeten geïnterpreteerd worden.

Nochtans valt het op dat bij zowel de fitness- als de precision-dimensie, de alignment based metrieken zich verschillend gedragen ten opzichte van de andere metrieken binnen hun respectievelijke dimensies. Er moet bij het interpreteren van deze metrieken rekening gehouden worden met hun gevoeligheid aan de kwaliteit van de modellen omdat ze wel degelijk een vertekend beeld kunnen vormen.

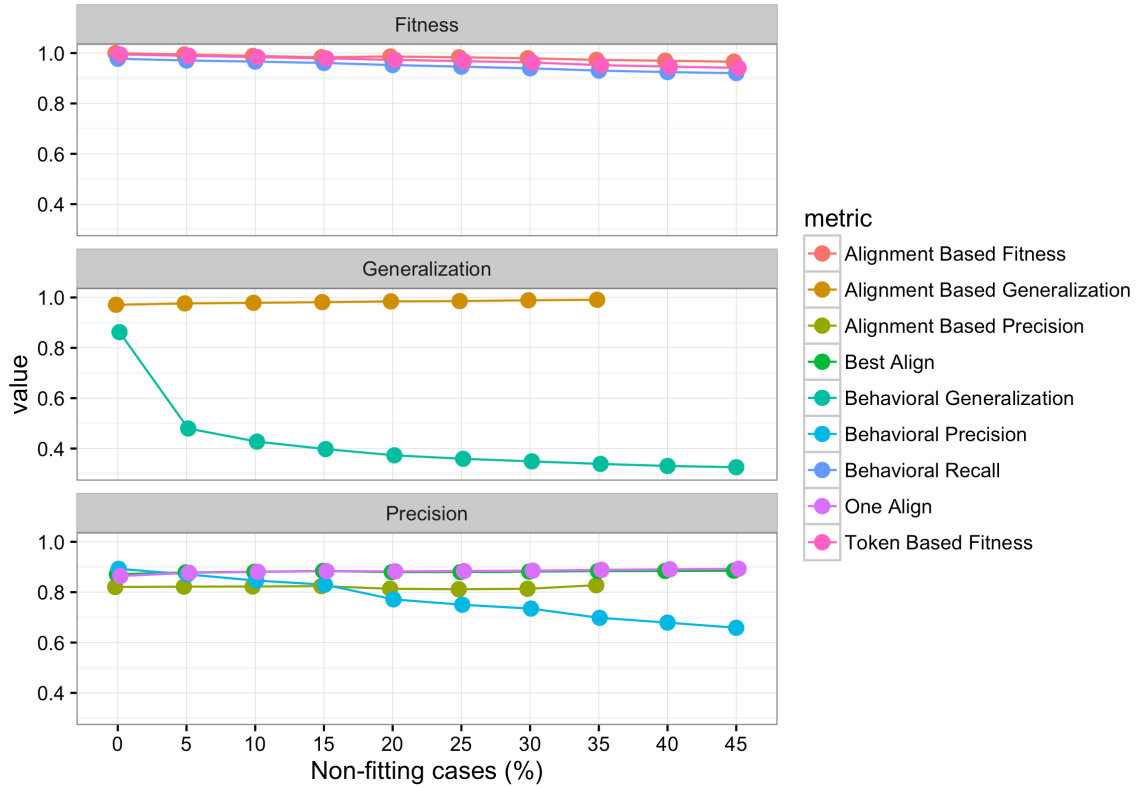
De geobserveerde metrieken die de dimensie generalization kwantificeren volgens de theorie, blijken dit in de praktijk niet te doen. Van Behavioral Generalization kan zelfs gezegd worden dat deze eerder de fitness-dimensie kwantificeert in geval dat de event log weinig noise bevat en redelijk compleet is. Over hoe deze metriek reageert op meer realistische modellen kan geen conclusie getrokken worden, aangezien er in de uitgevoerde analyses, ondanks een hogere complexiteit in experiment 1b, geen gebruik gemaakt werd van reële event logs. Over de Alignment Based Generalization kan gezegd worden dat deze correleert met geen enkele van de geobserveerde metrieken. Wat deze metriek dus praktisch gezien kwantificeert, is niet duidelijk waardoor het moeilijk is hier conclusies over te nemen. Op dit moment kan dus geconcludeerd worden dat er geen metrieken beschikbaar zijn die de generalization op een degelijke manier kwantificeren.

# Hoofdstuk 5: Analyse experiment 2

In dit hoofdstuk worden enkele analyses voor experiment 2 uitgevoerd en uitgelegd. In de eerste plaats zal de evolutie van de verschillende evaluatiemetrieken weergegeven worden over de verschillende niveaus van non-fitting cases. Vervolgens zal aan de hand van de Kruskal-Wallis test gekeken worden of er een significante invloed aanwezig is van het niveau van non-fitting cases op de waarde van de verschillende metrieken. Tot slot zal aan de hand van een regressieanalyse onderzocht worden wat de concrete invloed is van de verschillende niveaus van non-fitting cases op het niveau van de metrieken.

## 5.1 Algemene analyse

Op basis van de data in experiment 2, kan de volgende figuur opgesteld worden.




*Figuur 32: Invloed non-fitting cases op evaluatiemetrieken*

Deze figuur geeft de gemiddelde waarde van elke metriek weer, gegeven een bepaald niveau van non-fitting cases. Dat de fitness daalt naarmate het niveau van noise toeneemt wordt duidelijk op basis van de drie beschouwde fitness-metrieken. Nochtans dalen deze metriek slechts matig. Zelfs bij 45% non-fitting cases kwantificeren de geobserveerde metrieken de fitness nog steeds zeer hoog. Dit wijst erop dat ten opzichte van de naïeve fitness metriek[9] (aantal traces die correct afgespeeld kunnen worden), de geobserveerde metrieken de fitness zeer optimistisch kwantificeren. Dit resultaat werd eveneens gevonden in [44].

[9] Een waarde van 0,50 voor deze metriek kan geïnterpreteerd worden als een aanwezigheid van 50% non-fitting cases in de event log.

Als er gekeken wordt naar de precision-metrieken valt op dat het effect van het niveau van non-fitting cases niet zo groot is als eerst verwacht. Nochtans zijn wel duidelijke evoluties waar te nemen. Zo daalt de Behavioral Precision relatief sterk ten opzichte van de andere precision-metrieken. Op basis van deze figuur, kan gezegd worden dat deze metriek, binnen de dimensie precision, het meest gevoelig is aan het niveau van non-fitting cases. De Alignment Based Precision daarentegen blijft min of meer constant, wat tegen de verwachtingen in gaat, aangezien ook dit een alignment based metriek is. De laatste twee metrieken, One Align en Best Align Precision, nemen wel licht toe naarmate het noise-level stijgt maar de invloed blijkt slechts gering te zijn.

Als tenslotte de dimensie generalization beschouwd wordt, valt een duidelijke en sterke daling op van de Behavioral Generalization naarmate het level van de noise toeneemt. Het valt eveneens op dat van zodra er 5% non-fitting cases aanwezig zijn, een sterke daling wordt waargenomen in deze metriek. De metriek daalt nog verder naarmate het percentage non-fitting cases toeneemt, maar minder sterk. Hieruit kan dus besloten worden dat deze metriek zeer gevoelig is aan de hoeveelheid non-fitting traces in de log. Dit sluit aan bij het eerder gevonden resultaat dat deze metriek eerder de dimensie fitness kwantificeert. De Alignment Based Generalization neemt daarentegen licht toe en reageert dus op een geheel andere manier dan de Behavioral Generalization.

Merk op dat bij het opstellen van deze figuur vooraf gecontroleerd werd voor de invloed van zowel het systeem als het niveau van completeness. Voor beiden werd vastgesteld dat er geen significante invloed aanwezig was op het niveau van de metrieken en dat het effect op de metrieken dus enkel afkomstig is van het niveau van non-fitting cases.

Uit Figuur 32 kan een bepaalde invloed van non-fitting cases op de metrieken worden waargenomen, maar wat niet kan worden opgemaakt is of deze invloed ook significant is. Daarom wordt gekozen om de Kruskal-Wallis test uit te voeren [45]. Dit is een non-parametrische test die met andere woorden verdelingsvrij is. De gebruikte data is niet normaal verdeeld, waardoor de parametrische equivalente test, namelijk de ANOVA test, niet geldig gebruikt mag worden. De Kruskal-Wallis test gaat de data binnen een groep een rank toekennen waarmee vervolgens de verschillen tussen de groepen vergeleken worden. De groepen worden bepaald door het niveau van non-fitting cases. Er wordt voor elke metriek apart een Kruskal-Wallis test uitgevoerd die gaat weergeven of er een invloed is van non-fitting cases op deze respectievelijke metriek.

De nulhypothese die gebruikt werd stelt dat er geen verschillen zijn tussen de verschillende groepen (het niveau van non-fitting cases). Van zodra één groep aanwezig is die afwijkt van de anderen, zal deze test een significant verschil aangeven, waardoor deze nulhypothese verworpen zal worden. Met andere woorden, als een significante invloed van de non-fitting cases op een bepaalde metriek wordt waargenomen, moet de nulhypothese voor deze metriek verworpen worden.

De resultaten van deze test worden voor elke metriek samengevat in de volgende tabel:

| Dimensie | Metriek | Kruskal-Wallis chi-squared | p-waarde | |
|---|---|---|---|---|
| Fitness | Alignment Based Fitness | 1560,6 | < 2,2e-16 | *** |
| | Behavioral Recall | 545,62 | < 2,2e-16 | *** |
| | Token Based Fitness | 1016,5 | < 2,2e-16 | *** |
| Precision | One Align Precision | 16,665 | 0,05422 | . |
| | Best Align Precision | 4,3807 | 0,8846 | |
| | Behavioral Precision | 795,02 | < 2,2e-16 | *** |
| | Alignment Based Precision | 4,2573 | 0,7497 | |
| Generalization | Behavioral Generalization | 639,3 | < 2,2e-16 | *** |
| | Alignment Based Generalization | 42,538 | 4,096e-07 | *** |
| Significantieniveaus: 0,1% ***, 1% **, 5% *, 10% . | | | | |

*Tabel 13: Samenvatting van de Kruskall-Wallis test*

Uit deze test wordt duidelijk dat de meeste metrieken wel degelijk een significante invloed ondervinden van het niveau van non-fitting cases. Zowel de fitness- als de generalization-metrieken worden significant beïnvloedt door het niveau van non-fitting cases. Bij enkele van de precision-metrieken is deze invloed echter niet sterk aanwezig. Zo wordt bij de Best Align en de Alignment Based Precision geen significante invloed waargenomen. Van beide werd nochtans verwacht uit de theorie dat er een bepaalde invloed was. Bij de One Align en Behavioral Precision is er echter wel een significante invloed aanwezig, respectievelijk op het 10% en 0,1% significantieniveau (s.n.). Ook bij de One Align Precision is er dus een minder sterke invloed dan verwacht.

Op basis van deze analyse kan enkel vastgesteld worden of een bepaalde invloed van het niveau van non-fitting cases aanwezig is bij de verschillende metrieken. Wat echter niet duidelijk is, is de richting van deze invloed. Met andere woorden, er is niet geweten of het niveau van non-fitting cases deze metrieken positief of negatief beïnvloedt. Om dit te onderzoeken wordt voor elk van de metrieken waar een significante invloed van non-fitting cases gevonden wordt, een regressieanalyse uitgevoerd.

## 5.2 Regressieanalyse

Zoals reeds vermeld wordt er voor elke metriek waarbij een significante invloed gevonden werd, een regressieanalyse uitgevoerd. Zo kan gekeken worden op welke manier de non-fitting cases de waarde van de metrieken beïnvloedt. Naast een lineaire regressieanalyse werd ook getracht om een niet lineaire modellen te fitten met de data. Hieruit bleek echter dat enkel het lineaire model significant was voor de verschillende metrieken, met uitzondering van de Behavioral Generalization. Het gebruikte model voor de lineaire regressies is de volgende:

$$waarde\ metriek = \beta_0 + \beta_1 * (\%\ non - fitting\ cases) + \beta_2 * (completeness2) + \cdots + \beta_4 * (completeness4) + \beta_5 * (system10\_2) + \cdots + \beta_{13}(system9\_1)$$

De term non-fitting cases wordt uitgedrukt als een getal tussen 0 en 1, wat het percentage van non-fitting cases in de event log voorstelt. De resultaten moeten bijgevolg geïnterpreteerd worden als volgt: de waarde van de metriek wordt voorspeld als $\beta_0$ opgeteld met het product van $\beta_1$ en het niveau van non-fitting cases (uitgedrukt als percentage), de $\beta$ van het level van completeness dat werd toegoepast en de $\beta$ van het gebruikte systeem. Enkel voor de Behavioral Generlization werd dus een polynomiaal model van de tweede macht gebruikt dat beter fit met de data, namelijk:

$$waarde\ metriek = \beta_0 + \beta_1 * (\%\ non - fitting\ cases) + \beta_2 * (\%\ non - fitting\ cases)^2 + \beta_7 * (completeness2) + \cdots + \beta_9 * (completeness4) + \beta_{10} * (system10\_2) + \cdots + \beta_{18}(system9\_1)$$

De interpretatie is gelijkaardig, met het verschil dat de juiste macht moet toegepast worden op het niveau van non-fitting cases om tot een voorspelling van de waarde van de Behavioral Generalization te komen. De resultaten van deze analyse worden in een aparte tabel samengevat. Beide tabellen worden op de volgende pagina weergegeven. Merk op dat de vetgedrukte waarden niet meer significant zijn op het 5% s.n..

Alvorens de resultaten van de regressieanalyses geïnterpreteerd zullen worden, moet eerst een korte analyse gemaakt worden van de accuraa theid en kwaliteit van deze regressies. Om de accuraatheid van het regressiemodel te testen, wordt in de literatuur doorgaans gebruik gemaakt van de zogeheten *coefficient of determination* ($R^2$) [38]. Als de geöbserveerde waarde van de afhankelijke variabele wordt voorgesteld als $y_i$, $\bar{y}$ als het gemiddelde hiervan en $\hat{y}_i$ als de voorspelde waarde, dan wordt de $R^2$ berekend door volgende formule:

$$R^2 = \frac{\sum(\hat{y}_i - \bar{y})^2}{\sum(y_i - \bar{y})^2}$$

Het stelt de totale hoeveelheid variantie voor in de afhankelijke variabele (de metriek) die verklaard wordt door de onafhankelijke variabelen (niveau van non-fitting cases, completeness en systeem). Hoe hoger deze meeteenheid, hoe meer variantie in de metriek verklaard wordt door de opgenomen variabelen. Voor elk van de uitgevoerde analyses is deze maaststaf voldoende hoog, wat betekent dat de opgenoment onafhankelijke variabelen een groot deel van de variantie in de afhankelijke variabele verklaren. De interpretatie van deze maatstaf is als volgt: 93,5% van de variantie in de One Align Precision wordt verklaard door het niveau van non-fitting cases, het niveau van completeness en de keuze van het systeem. Enkel voor de Alignment Based Generalization is deze maatstaf lager, namelijk 0,54. Net zoals in het vorige experiment is dit een aanwijzing dat er een andere, niet opgenomen onafhankelijke variabele is die de variantie in de Alignment Based Generalization bepaald. Of met andere woorden, deze metriek wordt mede bepaald door een variabele die nog niet bekend is.

De residual error geeft vervolgens de standaardafwijking van de voorspelde waarden ten opzichte van de werkelijke waarden van de metrieken weer. Hoe kleiner deze waarde, hoe hoger de accuraatheid van het regressiemodel. Voor elk van deze analyses is de residual error relatief klein, wat bijgevolg wijst op een slechts geringe afwijking van de voorspelde waarden ten opzichte van de

werkelijke waarden van de metrieken. Er kan dus besloten worden dat de uitgevoerde regressies van een degelijke kwaliteit zijn.

| Coëfficiënten | Behavioral Generalization |
|---|---|
| Intercept | 0,92749 |
| non-fitting cases (%) | -2,70627 |
| non-fitting cases$^2$ | 4,26684 |
| completeness2 | 0,01915 |
| completeness3 | 0,04832 |
| completeness4 | 0,10897 |
| system10_2 | -0,24763 |
| system11_3 | -0,15831 |
| system2_3 | -0,30969 |
| system4_3 | -0,38455 |
| system5_5 | -0,23935 |
| system6_3 | -0,37056 |
| system7_7 | -0,30640 |
| system8_3 | -0,16934 |
| system9_1 | -0,23991 |
| R-squared ($R^2$) | 0,7236 |
| Residual error | 0,1111 |

*Tabel 14: Resultaat polynomiale regressie*

| Coëfficiënten | One Align | Behavioral Precision | Token Based Fitness | Alignment Based Fitness | Behavioral Recall | Alignment Based Generalization |
|---|---|---|---|---|---|---|
| Intercept | 0,91743 | 0,95709 | 0,97963 | 0,98319 | 0,94343 | 0,85458 |
| non-fitting cases (%) | 0,04662 | -0,59707 | -0,12313 | -0,06945 | -0,13175 | 0,06215 |
| completeness2 | -0,01561 | -0,00861 | **0,00010** | **-0,00011** | **-0,00008** | **-0,00482** |
| completeness3 | -0,03496 | -0,01885 | **0,00051** | **-0,00011** | **-0,00003** | -0,01710 |
| completeness4 | -0,07131 | -0,05315 | **-0,00020** | **-0,00011** | **-0,00034** | -0,04695 |
| system10_2 | 0,04298 | -0,02058 | 0,02443 | 0,01615 | 0,06593 | 0,15240 |
| system11_3 | -0,05108 | **0,00693** | **0,00037** | 0,01077 | 0,01685 | 0,14014 |
| system2_3 | 0,06466 | -0,07527 | 0,01991 | 0,01400 | 0,05818 | 0,15327 |
| system4_3 | 0,03320 | -0,09935 | 0,03211 | 0,02206 | 0,07144 | 0,15344 |
| system5_5 | -0,18246 | -0,18910 | -0,02717 | 0,01628 | -0,04208 | 0,15445 |
| system6_3 | **0,00195** | 0,02771 | 0,03306 | 0,01918 | 0,06606 | 0,15334 |
| system7_7 | 0,01512 | -0,11789 | 0,03197 | 0,01757 | **0,00024** | 0,15376 |
| system8_3 | 0,03384 | 0,01233 | 0,03444 | 0,01801 | 0,07186 | 0,15223 |
| system9_1 | -0,10953 | 0,05440 | 0,01641 | 0,01575 | 0,04328 | 0,13399 |
| R-squared ($R^2$) | 0,93593 | 0,77398 | 0,86716 | 0,83355 | 0,90402 | 0,54984 |
| Residual error | 0,02043 | 0,05791 | 0,01000 | 0,00515 | 0,01368 | 0,04504 |

*Tabel 15: Resultaten lineaire regressies*

Voor de interpretatie van de regressieanalyses wordt enkel gekeken naar de coëfficiënten van de term non-fitting cases. De reden voor het uitvoeren van deze regressieanalyses was namelijk om te onderzoeken op welke manier deze non-fitting cases de waarden van de metrieken beïnvloedden. Hoe zowel het niveau van completeness als het systeem deze waarden beïnvloeden is in dit experiment niet van belang. Deze termen werden enkel opgenomen in de analyse om een degelijke kwaliteit te bekomen.

Uit de opgestelde regressieanalyses wordt duidelijk dat het niveau van non-fitting cases een verschillende invloed heeft op elk van de metrieken. Voor de One Align Precision en de Alignment Based Generalization is dit een relatief lichte positieve invloed, terwijl dit voor de andere metrieken een negatieve invloed is. Met andere woorden, naarmate het niveau van non-fitting cases toeneemt, zal de One Align Precision en de Alignment Based Generalization toenemen, terwijl de waarde van de andere metrieken waarbij een significante invloed gevonden werd, zal afnemen. Wat opvalt is dat bij de One Align Precision een positieve invloed wordt waargenomen, terwijl dit bij de Behavioral Precision een negatieve invloed is. Het effect is dus niet zozeer dimensie-gebonden, maar eerder metriek-gebonden. Als het aantal non-fitting cases toeneemt met 5 procentpunten, dan zal bijvoorbeeld de Behavioral Precision afnemen met 0,0298 (0,59707 * 0,05). Voor de Behavioral Generalization blijkt dat indien het aantal non-fitting cases laag is, deze een negatieve invloed heeft op deze metriek. Naarmate het aantal non-fitting cases toeneemt, draait dit effect plots om en neemt het vervolgens exponentieel toe.

Het effect van de non-fitting cases is, ondanks het feit dat het een significante invloed is, slechts gering op de One Align Precision, de Alignment Based Fitness, Behavioral Recall en de Alignment Based Generalization. Terwijl dit effect op de Behavioral Precision en de Token Based Fitness veel sterker is. Het is belangrijk rekening te houden met deze effecten bij het interpreteren van de verschillende metrieken, aangezien elke metriek anders reageert op het niveau van non-fitting cases.

## 5.3 Conclusie

In de literatuur werd een contradictie gevonden met betrekking tot de invloed van non-fitting cases op het niveau van precision. Er werd namelijk, op basis van het Venn diagram in sectie 2.1, vastgesteld dat de dimensies fitness, die mede bepaald wordt door het niveau van non-fitting cases, en precision onafhankelijk moeten zijn van elkaar. Nochtans zijn er enkele metrieken die deze onafhankelijk lijken te negeren, zoals besproken werd in sectie 3.2. Daarom werden in dit hoofdstuk enkele analyses uitgevoerd om de invloed van non-fitting cases op de verschillende metrieken te onderzoeken. Er werd verwacht dat de Alignment Based, One Align en Best Align Precision zouden toenemen naarmate het niveau van non-fitting cases toeneemt. Van de fitness-metrieken werd verwacht dat deze dalen naarmate het niveau van non-fitting cases toeneemt. De invloed van deze non-fitting cases op de metrieken die de dimensie generalisation kwantificeren is tot slot moeilijk te voorspellen.

Uit de uitgevoerde analyses in dit hoofdstuk blijkt er een positieve invloed van het niveau van non-fitting cases te zijn op de waarde van One Align Precision, wat een bevestiging is van de verwachting. Nochtans, voor de metrieken Alignment Based en Best Align Precision werd de verwachting niet voldaan. Er werd namelijk geen significante invloed gevonden op zowel de Alignment Based als de Best Align Precision.

Voor de dimensie fitness werden wel de verwachte resultaten gevonden, namelijk een significante daling van elk van deze metrieken naarmate het niveau van non-fitting cases toeneemt. Nochtans is het effect hiervan slechts gering wat zou kunnen wijzen op het feit dat de huidige geobserveerde metrieken de fitness te positief kwantificeren.

In geval van de dimensie generalization werden gemengde resultaten gevonden. De Alignment Based Generalization ondervindt namelijk een relatief licht positieve invloed van het aantal non-fitting cases, terwijl de Behavioral Generalization in geval van lage niveaus van non-fitting cases negatief beïnvloed wordt. Naarmate het niveau van non-fitting cases toeneemt draait dit effect plots om, en neemt vervolgens exponentieel toe.

# Hoofdstuk 6: Conclusies

Dit werkstuk geeft een overzicht van enkele van de meest gehanteerde metrieken om de kwaliteit van procesmodellen te kwantificeren. Er werden metrieken besproken voor de dimensies fitness, precision en generalization. Het werd duidelijk dat elk van de metrieken een verschillende methode gebruikt om zijn respectievelijke dimensie te kwantificeren, vaak met behulp van complexe formules. Hierdoor is het moeilijk om een duidelijke interpretatie te vormen van de gevonden waarden waardoor het relatief vergelijken van verschillende modellen wel mogelijk is, maar het moeilijk is om de kwaliteit van een enkel model in te schatten. Bovendien is het niet duidelijk of de besproken metrieken wel degelijk hun respectievelijke dimensie kwantificeren. Hiernaast is het niet geweten of elk van de metrieken op een gelijke manier reageert op een verandering van de context (bijvoorbeeld indien er gebruik gemaakt wordt van een ander discovery algoritme).

De experimenten die opgezet werden proberen nieuwe inzichten te verwerven over deze metrieken. De analyses uit experiment 1 maken duidelijk dat de geobserveerde fitness-metrieken deze dimensie op een degelijke manier kwantificeren. Token Based Fitness en Behavioral Recall gedragen zich bij zowel modellen van hoge als lage kwaliteit zeer gelijkaardig. Nochtans heeft de Token Based Fitness meer capaciteit nodig dan de Behavioral Recall om tot een oplossing te komen. De laatste fitness-metriek, Alignment Based Fitness, gedraagt zich in geval van modellen van een hoge kwaliteit gelijkaardig aan de andere twee fitness-metrieken, maar wijkt van deze metrieken af naarmate de kwaliteit van de ontdekte modellen daalt. Deze metriek vertoont in die gevallen een afvlakkend effect, waardoor deze metriek een vertekend beeld geeft over de fitness. Deze metriek is in vergelijking met de andere twee, dus minder betrouwbaar. Voor het kwantificeren van de dimensie fitness kan best gebruik gemaakt worden van de Token Based Fitness of de Behavioral Recall. Rekening houdende met de benodigde rekencapaciteit, lijkt de Behavioral Recall nochtans de beste optie te zijn.

Als vervolgens de metrieken beschouwd worden die de dimensie precision kwantificeren, blijkt dat drie van de vier geobserveerde metrieken zich zeer gelijkaardig gedragen. Zowel de One Align, Best Align als de Behavioral Precision kwantificeren de precision op een consistente manier voor zowel modellen van hoge als lage kwaliteit. De Alignment Based Precision is echter minder betrouwbaar om de precision de kwantificeren omdat deze afwijkend gedrag vertoont gegeven bepaalde niveau's van precision. De Best Align vertoont in geval van zeer hoge precision van het model afvlakkend en uiteindelijke zelfs dalend gedrag. Hierdoor is ook deze metriek minder betrouwbaar om te hanteren. Er kan dus best gebruik gemaakt worden van ofwel de One Align of de Behavioral Precision, omdat beide metrieken de precision op een degelijke manier lijken te kwantificeren. Nochtans hebben zowel de Best Align als de One Align Precision zeer veel rekentijd nodig om tot een oplossing te komen, waardoor de Behavioral Precision het meest aangewezen wordt om de precision te kwantificeren.

Over de metrieken die de laatste dimensie, generalization, kwantificeren volgens de literatuur kunnen zeer moeilijk conclusies genomen worden. Deze dimensie kwam namelijk niet duidelijk naar voor in de data. De Behavioral Generalization blijkt zelfs eerder de dimensie fitness te kwantificeren aangezien deze metriek in geval van zowel modellen van hoge en lage kwaliteit sterk correleert met

de fitness-metrieken. De tweede generalization-metriek, Alignment Based Generalization, vertoont geen enkel verband met eender andere geobserveerde metriek. Hierdoor kan er over deze metriek geen conclusie genomen worden. Wat echter wel geconcludeerd kan worden, is dat er op dit moment geen metrieken beschikbaar zijn die deze dimensie op een betrouwbare en consistente manier kwantificeren.

Het tweede uitgevoerde experiment geeft meer inzicht over hoe onafhankelijk de fitness en precision zijn van elkaar. De theorie gaf namelijk aan dat er geen correlatie aanwezig zou mogen zijn tussen deze twee dimensies, terwijl er enkele precision-metrieken deze onafhankelijkheid blijken te negeren. Uit de uitgevoerde analyses wordt duidelijk dat er geen significante invloed is van het niveau van non-fitting cases op zowel de Best Align als de Alignment Based Precision. Voor de One Align en Behavioral Precision is er echter wel een significante invloed gevonden, respectievelijk een positieve en negatieve invloed. De invloed op de One Align Precision is nochtans niet zeer sterk. Desalniettemin, is dit een aanwijzing dat de onafhankelijkheid tussen de fitness en precision niet in elke metriek wordt gehandhaafd.

Naast de precision-metrieken, werd ook de invloed van de non-fitting cases op de andere metrieken onderzocht. Zo bleek dat er een negatieve significante invloed aanwezig is op zowel de Token Based en Alignment Based Fitness als de Behavioral Recall. Dit resultaat werd eveneens verwacht, ook al is het effect veel minder sterk dan oorspronkelijk gedacht. Dit wijst erop dat de geobserveerde fitness-metrieken deze dimensie positiever weergeven dan de werkelijkheid. Voor beide geobserveerde generalization-metrieken werd eveneens een significante invloed gevonden. Voor de Alignment Based Generalization was deze invloed slechts gering, terwijl de Behavioral Generalization zeer gevoelig is aan het niveau van non-fitting cases. Dit wijst erop dat, net zoals in het eerste experiment, deze metriek eerder de dimensie fitness kwantificeert.

Ondanks dat dit onderzoek ettelijke nieuwe inzichten heeft verworven in de verschillende beschikbare metrieken, zijn er ook enkele nieuwe vragen naar voor gekomen. In de eerste plaats wordt duidelijk dat er nog veel onderzoek moet gebeuren naar welke metrieken op een betrouwbare manier de dimensie generalization kunnen kwantificeren. Er zijn op dit moment geen metrieken beschikbaar die in staat zijn dit te doen, waardoor het nog niet mogelijk is om een duidelijk beeld te vormen van deze dimensie.

Daarnaast moet er onderzocht worden op welke manier de gevonden waarden van de verschillende metrieken geïnterpreteerd moeten worden. Een bekomen waarde voor één bepaalde metriek kan wel relatief gebruikt worden om modellen onderling te vergelijken, maar het geeft geen duidelijk beeld over hoe hoog of laag de kwaliteit van een enkel model is. Daarom moet er vastgelegd worden vanaf welke waarde men spreekt over een model van goede kwaliteit en dit voor elke metriek afzonderlijk. Uit het uitgevoerde onderzoek werd duidelijk dat er voor de dimensies fitness en precision metrieken beschikbaar zijn die hun respectievelijke dimensie op een gelijkaardige manier kwantificeren, maar waarbij het niveau van de gevonden waarden consistent verschillend is. Het is dus niet zo dat bijvoorbeeld een waarde van 0,6 op de ene metriek, even goed is als een waarde van 0,6 op de andere metriek.

Een laatste voorstel tot verder onderzoek handelt over de gevonden significante invloed van non-fitting cases op enkele van de precision-metrieken. Uit de literatuur zou deze afhankelijkheid niet mogen voorkomen, waardoor er onderzocht moet worden welke invloed dit effect heeft op de interpretatie van de desbetreffende metrieken. Dit is belangrijk aangezien deze invloed een vertekend beeld kan vormen over de gevonden waarden van de metrieken.

Tot slot kan het uitgevoerde onderzoek toegepast worden op reële event logs, om zo na te gaan of de gevonden resultaten standhouden in de realiteit. Er werd getracht de realiteit zo goed mogelijk na te bootsen, maar het onderzoek geeft geen zekerheid over hoe goed de gegenereerde data overeenkomt met de realiteit. Om deze tekortkoming van dit onderzoek tegen te gaan, kunnen reële event logs verzameld worden om vervolgens het onderzoek opnieuw uit te voeren en zo de gevonden resultaten te bevestigen of te verwerpen.

# Bibliografie


[1] W. M. P. van der Aalst and A. K. Alves de Medeiros, "Process mining and security: Detecting anomalous process executions and checking process conformance," *Electron. Notes Theor. Comput. Sci.*, vol. 121, no. SPEC. ISS., pp. 3–21, 2005.

[2] W. M. P. van der Aalst, *Process Mining: Discovery, Conformance and Enhancement of Business Processes*, 1st ed. Berlin: Springer-Verlag Berlin Heidelberg, 2011.

[3] R. Škrinjar, V. Bosilj-Vukšic, and M. Indihar-Štemberger, "The impact of business process orientation on financial and non-financial performance," *Bus. Process Manag. J.*, vol. 14, no. 5, pp. 738–754, 2008.

[4] W. M. P. van der Aalst, A. Adriansyah, and B. Van Dongen, "Replaying history on process models for conformance checking and performance analysis," *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, vol. 2, no. 2, pp. 182–192, 2012.

[5] J. De Weerdt, M. De Backer, J. Vanthienen, and B. Baesens, "A multi-dimensional quality assessment of state-of-the-art process discovery algorithms using real-life event logs," *Inf. Syst.*, vol. 37, no. 7, pp. 654–676, 2012.

[6] W. M. P. van der Aalst, "Business alignment: Using process mining as a tool for Delta analysis and conformance testing," *Requir. Eng.*, vol. 10, no. 3, pp. 198–211, 2005.

[7] W. M. P. van der Aalst, T. Weijters, and L. Maruster, "Workflow mining: Discovering process models from event logs," *IEEE Trans. Knowl. Data Eng.*, vol. 16, no. 9, pp. 1128–1142, 2004.

[8] A. Rozinat and W. M. P. van der Aalst, "Conformance checking of processes based on monitoring real behavior," *Inf. Syst.*, vol. 33, no. 1, pp. 64–95, 2008.

[9] T. Jouck and B. Depaire, "Een verbeterde methode om process discovery technieken te vergelijken," Universiteit Hasselt, 2014.

[10] A. Rozinat, A. K. Alves de Medeiros, C. W. Günther, A. J. M. M. Weijters, and W. M. P. van der Aalst, "The need for a process mining evaluation framework in research and practice: position paper," *Proc. 2007 Int. Conf. Bus. Process Manag.*, pp. 84–89, 2008.

[11] S. K. L. M. Vanden Broucke, J. De Weerdt, J. Vanthienen, and B. Baesens, "A comprehensive benchmarking framework (CoBeFra) for conformance analysis between procedural process models and event logs in ProM," *Proc. 2013 IEEE Symp. Comput. Intell. Data Mining, CIDM 2013 - 2013 IEEE Symp. Ser. Comput. Intell. SSCI 2013*, pp. 254–261, 2013.

[12] S. K. L. M. Vanden Broucke, J. De Weerdt, J. Vanthienen, and B. Baesens, "Determining process model precision and generalization with weighted artificial negative events," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 8, pp. 1877–1889, 2014.

[13] J. C. a M. Buijs, B. F. Van Dongen, and W. M. P. van der Aalst, "On the role of fitness, precision, generalization and simplicity in process discovery," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 7565 LNCS, no. PART 1, pp. 305–322, 2012.

[14] J. C. a. M. Buijs, B. F. van Dongen, and W. M. P. van der Aalst, "Quality Dimensions in Process Discovery: The Importance of Fitness, Precision, Generalization and Simplicity," *Int. J. Coop. Inf. Syst.*, vol. 23, no. 1, p. 1440001, 2014.

[15] M. Weidlich, A. Polyvyanyy, N. Desai, J. Mendling, and M. Weske, "Process compliance analysis based on behavioural profiles," *Inf. Syst.*, vol. 36, no. 7, pp. 1009–1025, 2011.

[16] A. Rozinat, M. Veloso, and W. M. P. van der Aalst, "Evaluating the Quality of Discovered Process Models," *Inf. Syst. J.*, vol. 16, no. Section 2, pp. 1–8, 2008.

[17] W. M. P. van der Aalst, "Process Mining in the Large: A Tutorial," *Bus. Intell.*, pp. 33–76, 2014.

[18] J. Muñoz-Gama and J. Carmona, "A fresh look at precision in process conformance," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 6336 LNCS, pp. 211–226, 2010.

[19] A. Adriansyah, B. F. Van Dongen, and W. M. P. van der Aalst, "Conformance checking using cost-based fitness analysis," *Proc. - IEEE Int. Enterp. Distrib. Object Comput. Work. EDOC*, pp. 55–64, 2011.

[20] W. M. P. van der Aalst, "Mediating between modeled and observed behavior: The quest for the 'right' process: Keynote," *Proc. - Int. Conf. Res. Challenges Inf. Sci.*, 2013.

[21] W. M. P. van der Aalst, V. Rubin, H. M. W. Verbeek, B. F. Van Dongen, E. Kindler, and C. W. Günther, "Process mining: A two-step approach to balance between underfitting and overfitting," *Softw. Syst. Model.*, vol. 9, no. 1, pp. 87–111, 2010.

[22] J. C. A. M. Buijs, "Flexible Evolutionary Algorithms for Mining Structured Process Models," 2014.

[23] S. Goedertier, D. Martens, J. Vanthienen, and B. Baesens, "Robust process discovery with artificial negative events," vol. 10, pp. 1305–1340, 2009.

[24] S. Vanden Broucke, "Advances in Process Mining," Koninklijke Universiteit Leuven, 2014.

[25] J. De Weerdt, M. De Backer, J. Vanthienen, and B. Baesens, "A robust F-measure for evaluating discovered process models," *IEEE SSCI 2011 Symp. Ser. Comput. Intell. - CIDM 2011 2011 IEEE Symp. Comput. Intell. Data Min.*, pp. 148–155, 2011.

[26] J. Munoz-Gama and J. Carmona, "Enhancing precision in Process Conformance: Stability, confidence and severity," *IEEE SSCI 2011 Symp. Ser. Comput. Intell. - CIDM 2011 2011 IEEE Symp. Comput. Intell. Data Min.*, pp. 184–191, 2011.

[27] A. Adriansyah, J. Carmona, and B. F. Van Dongen, "Alignment Based Precision Checking," *Work. Bus. Process Intell. (BPI 2012)*, no. 1, pp. 1–12, 2012.

[28] A. Adriansyah, "Aligning Observed and Modeled Behavior," Technische Universiteit Eindhoven, Eindhoven, 2014.

[29] A. Vahedian Khezerlou and S. Alizadeh, "A new model for discovering process trees from event logs," *Appl. Intell.*, vol. 41, no. 3, pp. 725–735, 2014.

[30] T. Jouck and B. Depaire, "Generating Artificial Event Logs with Sufficient Discriminatory Power to Compare Process Discovery Techniques," pp. 1–5.

[31] a. J. M. M. Weijters, W. M. P. Van Der Aalst, and A. K. Alves de Medeiros, "Process Mining with the HeuristicsMiner Algorithm," *Cirp Ann. Technol.*, vol. 166, pp. 1–34, 2006.

[32] G. Janssenswillen, B. Depaire, and T. Jouck, "Calculating the Number of Unique Paths in a Block-Structured Process Model."

[33] C. Baier and J.-P. Katoen, *Principles Of Model Checking*, vol. 950. 2008.

[34] H. J. Cheng and A. Kumar, "Process mining on noisy logs - Can log sanitization help to improve performance?," *Decis. Support Syst.*, vol. 79, pp. 138–149, 2015.

[35] A. K. Alves de Medeiros, "Genetic process mining," 2006.

[36] J. M. E. M. van der Werf, B. F. van Dongen, C. A. J. Hurkens, and A. Serebrenik, "Process

discovery using integer linear programming," *Appl. Theory Petri Nets*, pp. 368–387, 2008.

[37] S. J. J. Leemans, D. Fahland, and W. M. P. van der Aalst, "Discovering block-structured process models from event logs containing infrequent behaviour," *Lect. Notes Bus. Inf. Process.*, vol. 171 LNBIP, pp. 66–78, 2014.

[38] J. F. Hair, W. C. Black, B. J. Babin, and R. E. Anderson, *Multivariate Data Analysis*, 7th ed. Pearson, 2014.

[39] D. D. Suhr, "Exploratory or Confirmatory Factor Analysis?," *Proc. 31st Annu. SAS Users Gr. Int. Conf.*, pp. 1–17, 2006.

[40] H. F. KAISER, "On Cliff's formula, the Kaiser-Guttman Rule, and the number of factors," *Percept. Mot. Skills*, vol. 74, no. 2, pp. 595–598, Apr. 1992.

[41] K. J. Preacher and R. C. MacCallum, "Exploratory factor analysis in behavior genetics research: factor recovery with small sample sizes," *Behav Genet*, vol. 32, no. 2, pp. 153–161, 2002.

[42] M. S. Bartlett, "Tests of Significance in Factor Analysis," *Br. J. Stat. Psychol.*, vol. 3, no. 2, pp. 77–85, 1950.

[43] A. Gie Yong and S. Pearce, "A Beginner's Guide to Factor Analysis: Focusing on Exploratory Factor Analysis," *Tutor. Quant. Methods Psychol.*, vol. 9, no. 2, pp. 79–94, 2013.

[44] J. De Weerdt, M. De Backer, J. Vanthienen, and B. Baesens, "A critical evaluation study of model-log metrics in process discovery," *Lecture Notes in Business Information Processing*, vol. 66 LNBIP. pp. 158–169, 2011.

[45] W. H. Kruskal and W. A. Wallis, "Use of Ranks in One-Criterion Variance Analysis," *J. Am. Stat. Assoc.*, vol. 47, no. 260, pp. 583–621, 1952.

[46] M. De Leoni, W. M. P. van der Aalst, and B. F. Van Dongen, "Data- and resource-aware conformance checking of business processes," *Lect. Notes Bus. Inf. Process.*, vol. 117 LNBIP, pp. 48–59, 2012.

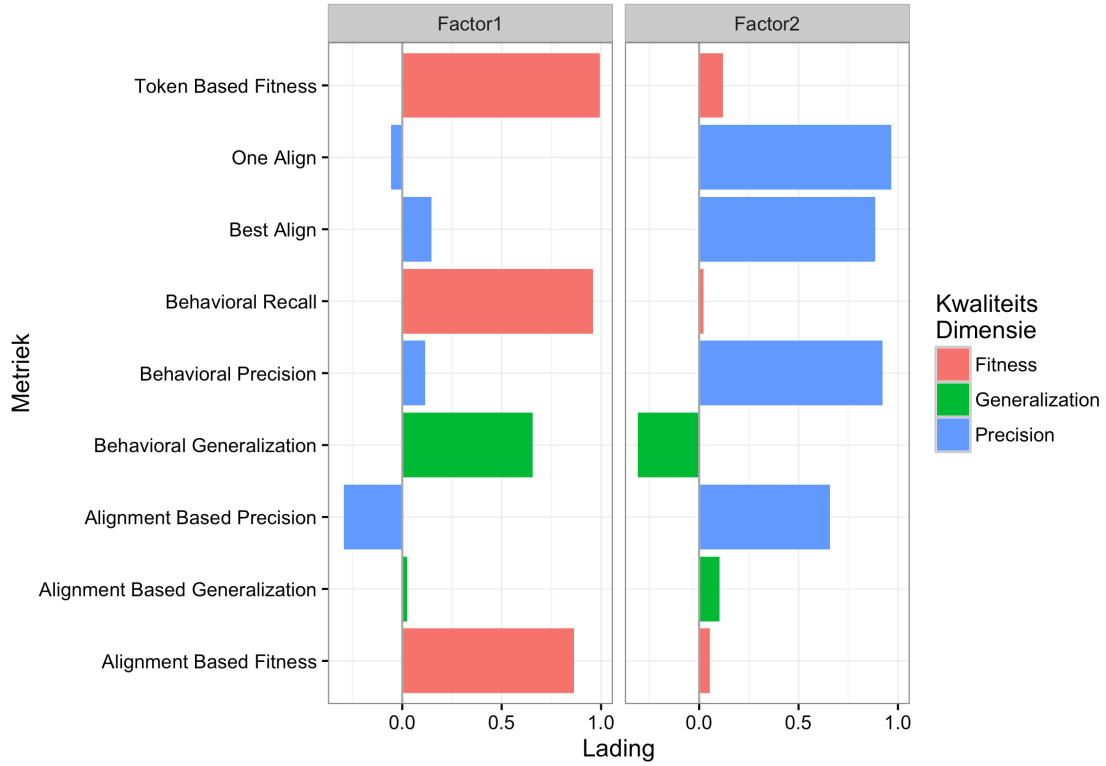
# Lijst van figuren

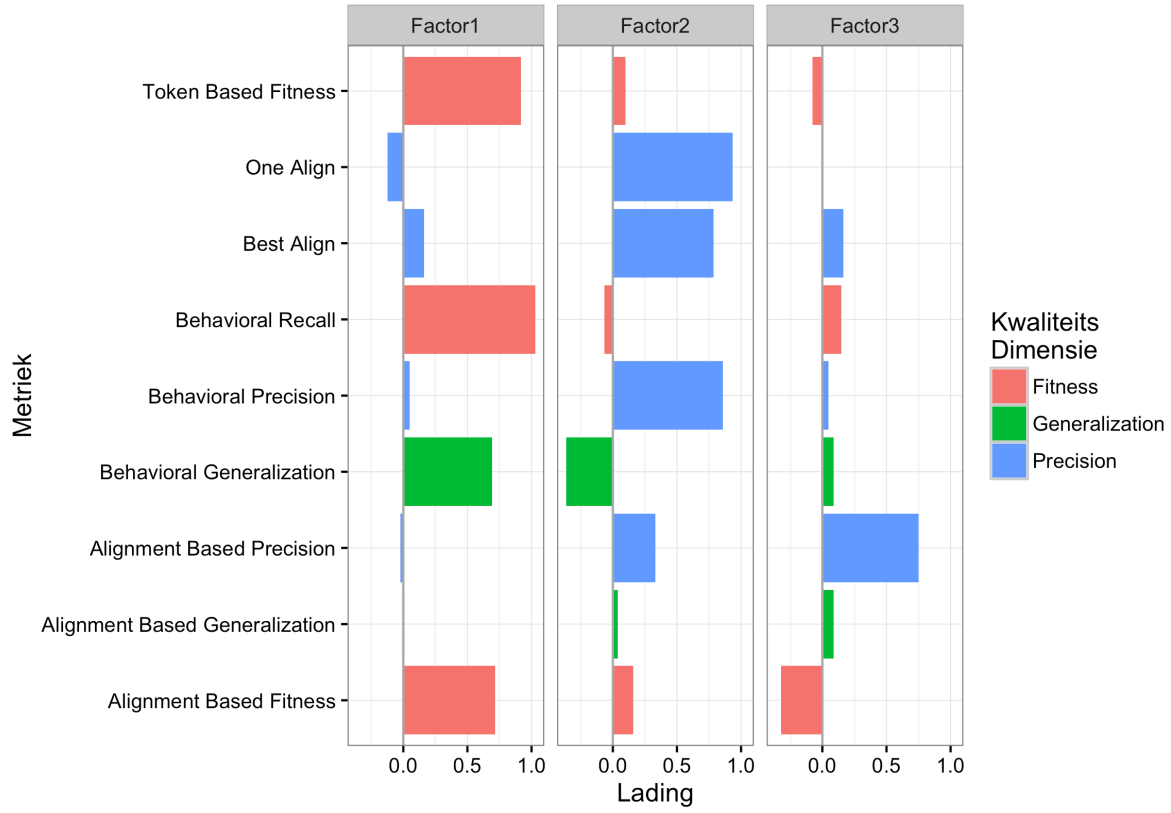
# Lijst van tabellen

# Bijlagen

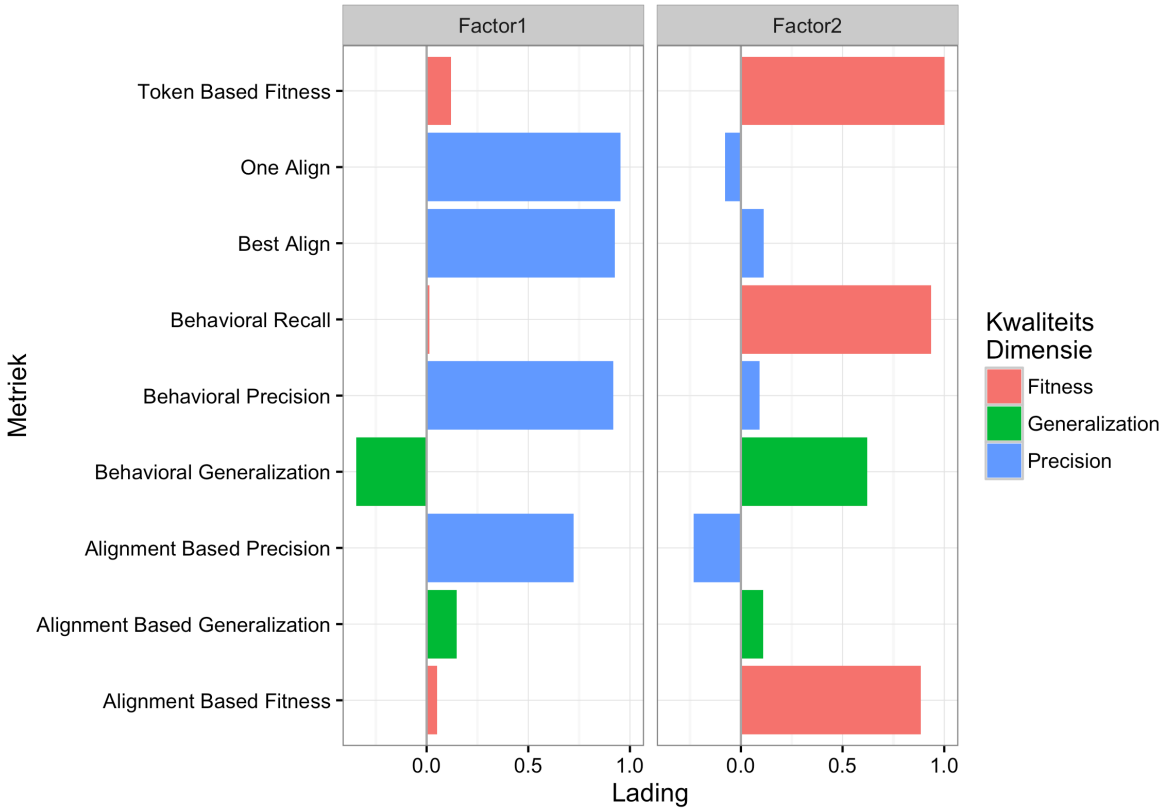
## Bijlage A

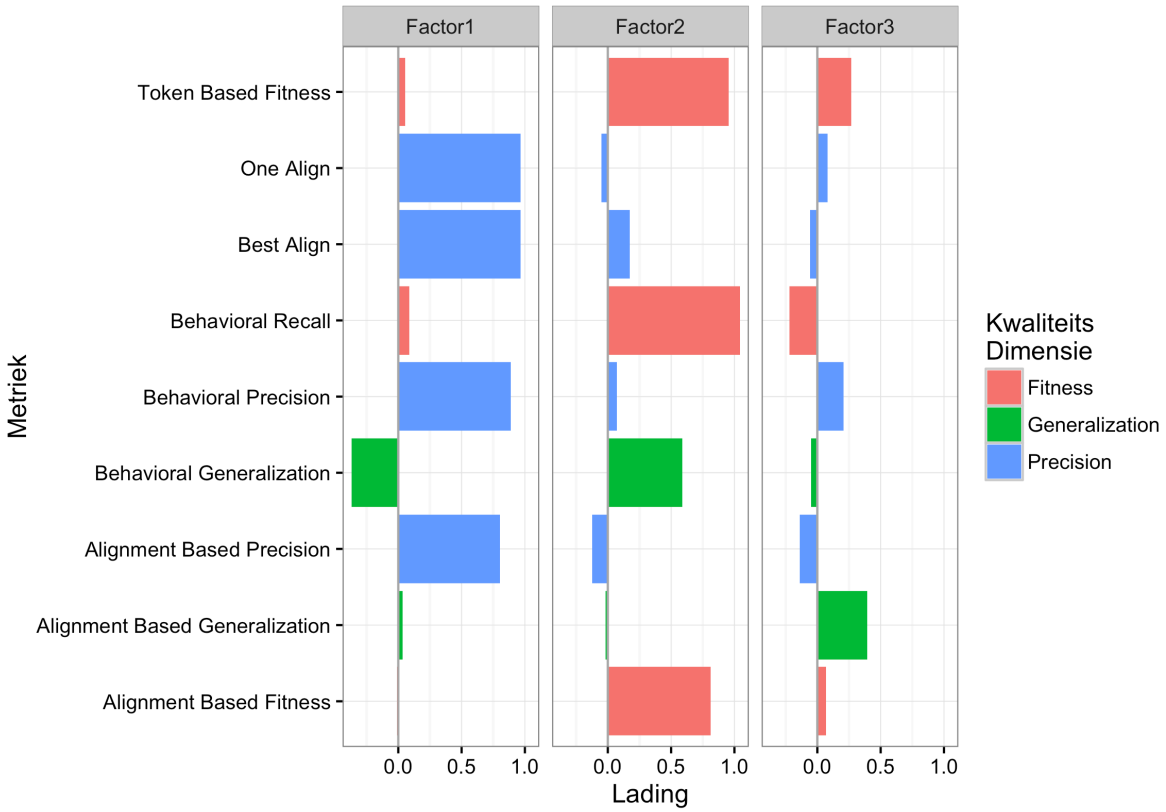
### Factoranalyses experiment 1

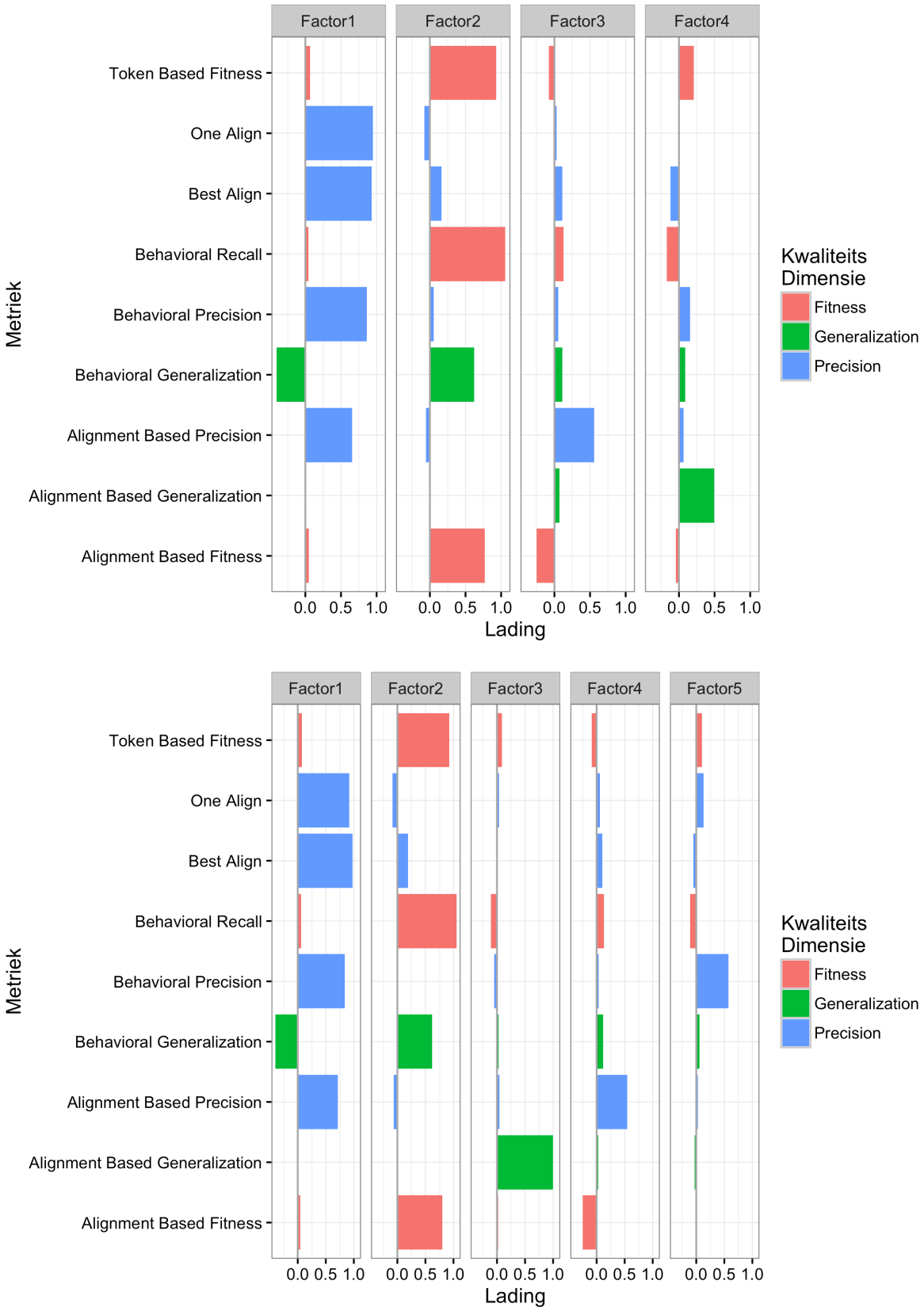


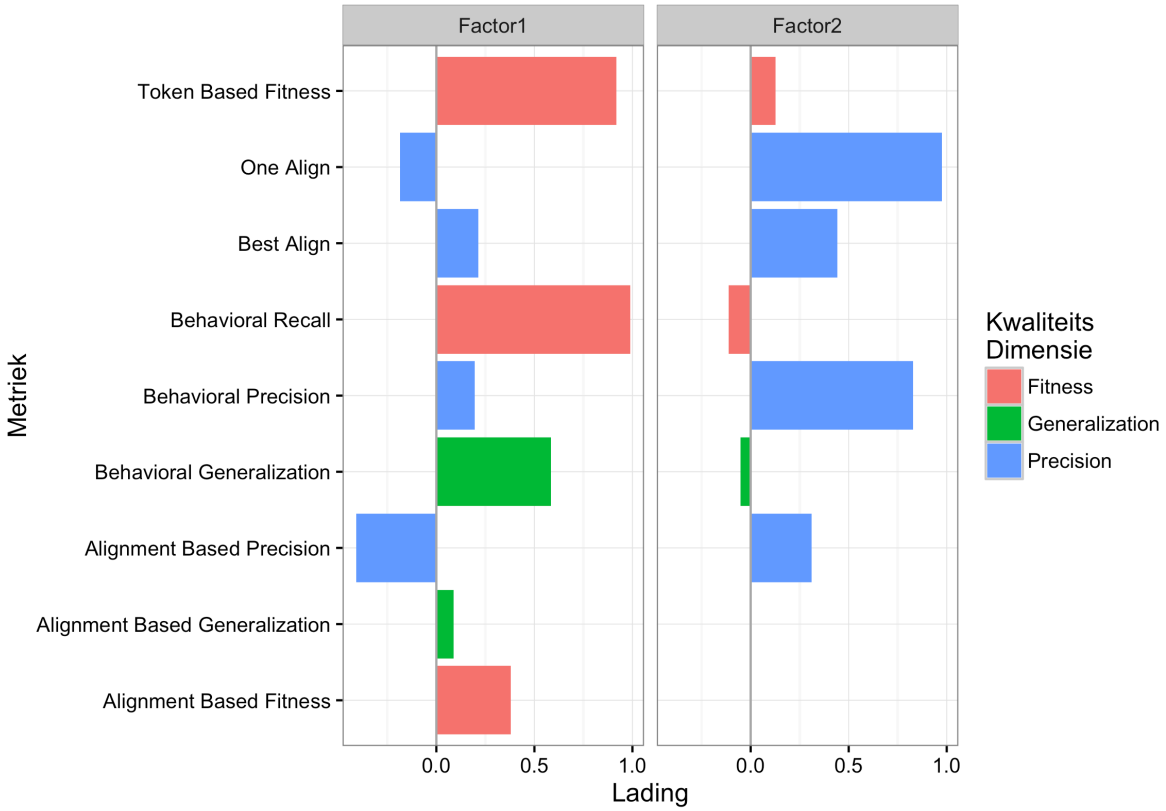
## Factoranalyses experiment 1a

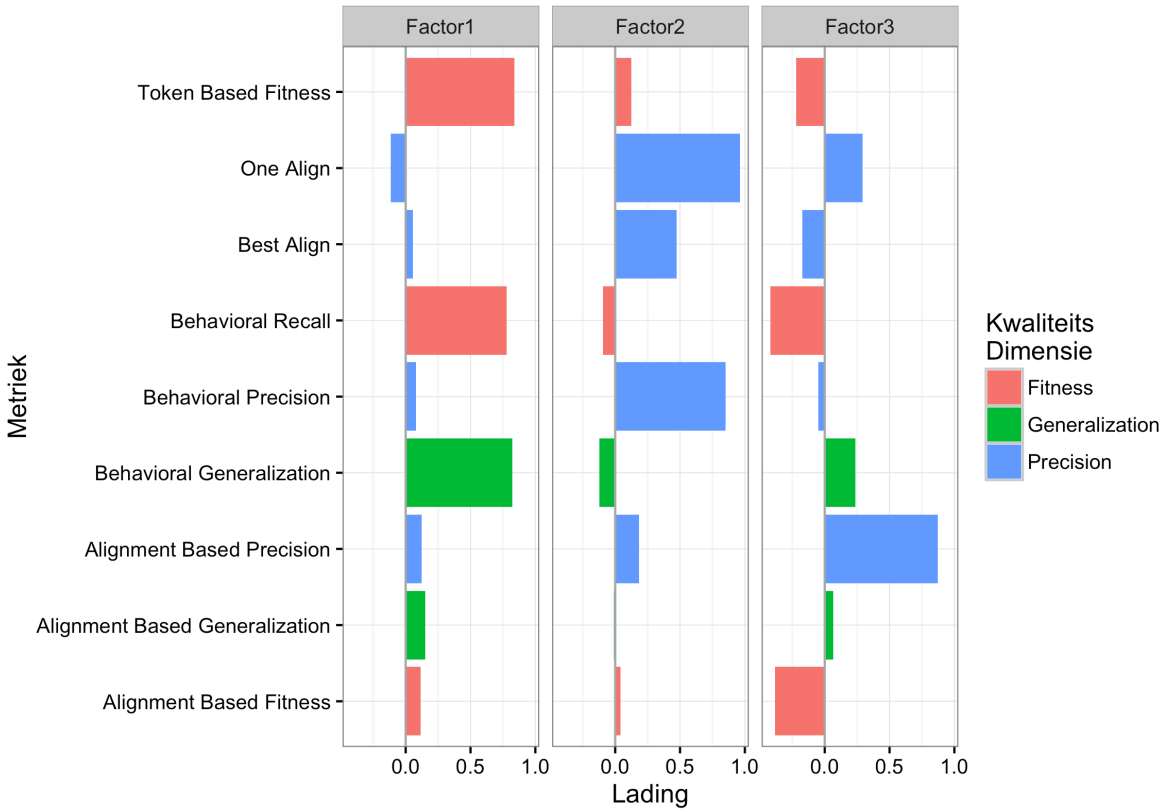


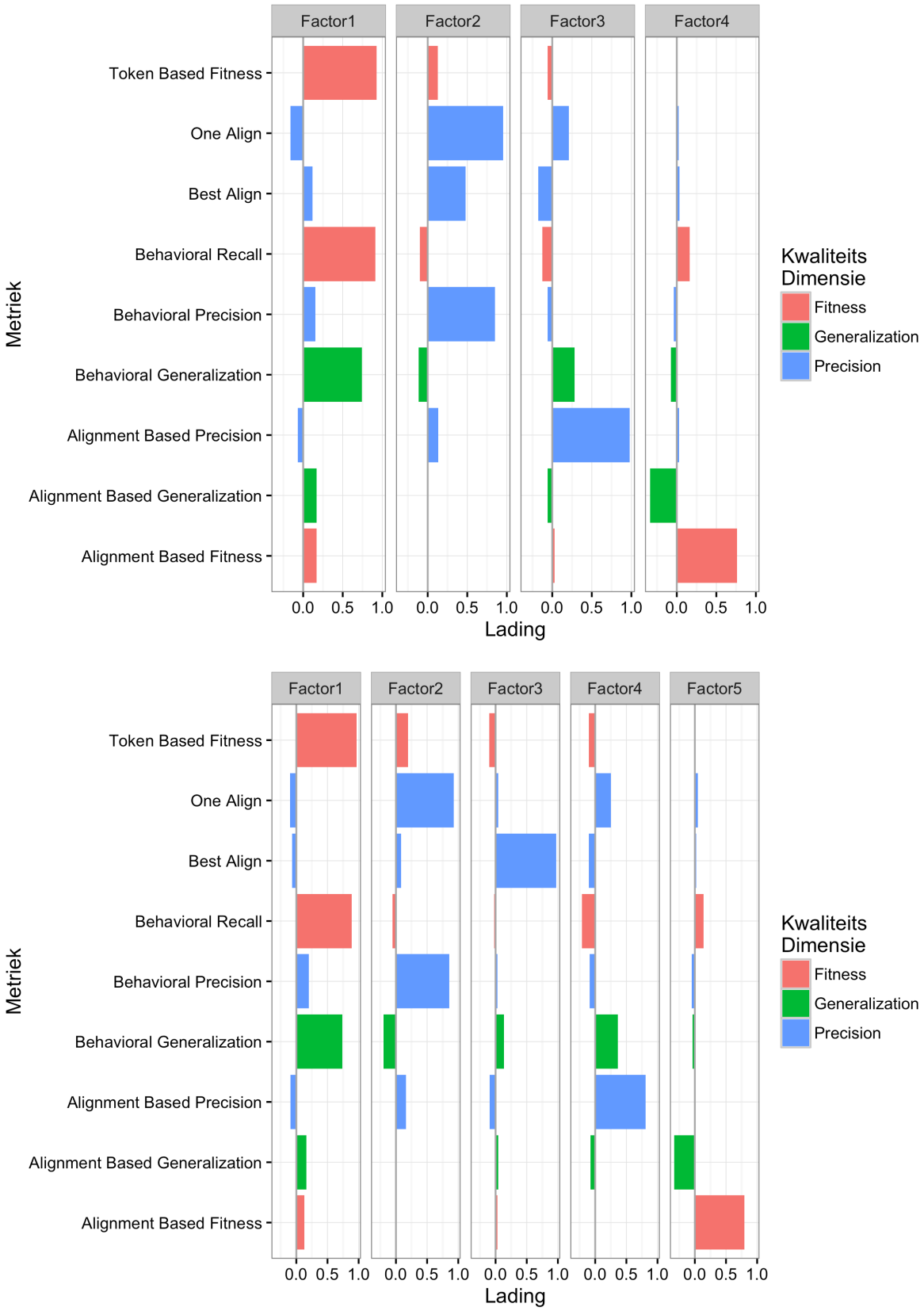




Factor1
Factor2
Factor3
Factor4
Factor5
Token Based Fitness
One Align
Best Align
Behavioral Recall
Behavioral Precision
Behavioral Generalization
Alignment Based Precision
Alignment Based Generalization
Alignment Based Fitness
Metriek
0.0 0.5 1.0
Lading
Kwaliteits Dimensie
Fitness
Generalization
Precision

**Factoranalyses experiment 1b**




Factor1
Factor2
Factor3
Factor4
Factor5
Token Based Fitness
One Align
Best Align
Behavioral Recall
Behavioral Precision
Behavioral Generalization
Alignment Based Precision
Alignment Based Generalization
Alignment Based Fitness
Metriek
0.0 0.5 1.0
Lading
Kwaliteits Dimensie
Fitness
Generalization
Precision

## Kwaliteit factoranalyses experiment 1

| | Communaliteit | | |
|---|---|---|---|
| Metriek | 2 Factoren | 3 Factoren | MSA |
| Alignment Based Fitness | 0,70254 | 0,76827 | 0,80018 |
| Alignment Based Precision | 0,72039 | 0,87021 | 0,76798 |
| Best Align | 0,66892 | 0,69630 | 0,87443 |
| Behavioral Generalization | 0,73603 | 0,72802 | 0,91608 |
| Behavioral Precision | 0,75170 | 0,76232 | 0,76383 |
| Behavioral Recall | 0,89596 | 0,95180 | 0,68228 |
| Token Based Fitness | 0,87430 | 0,86918 | 0,69600 |
| Alignment Based Generalization | 0,00829 | 0,01862 | 0,05993 |
| One Align | 0,99284 | 0,98060 | 0,78688 |

| | | | |
|---|---|---|---|
| Totale Communaliteit | 0,70566 | 0,73836 | KMO |
| Bartlett's p-waarde | 0,00000 | 0,00000 | 0,76088 |
| RMSR | 0,03879 | 0,03351 | |

## Kwaliteit factoranalyses experiment 1a en 1b

Experiment 1a

| | Communaliteit | | | | |
|---|---|---|---|---|---|
| Metriek | 2 Factoren | 3 Factoren | 4 Factoren | 5 Factoren | MSA |
| Alignment Based Fitness | 0,74024 | 0,79437 | 0,75494 | 0,77822 | 0,82291 |
| Alignment Based Generalization | 0,01681 | 0,02419 | 0,25384 | 0,99500 | 0,05015 |
| Alignment Based Precision | 0,76074 | 0,90680 | 0,99501 | 0,99501 | 0,76665 |
| Best Align | 0,75669 | 0,77638 | 0,77879 | 0,83178 | 0,85673 |
| Behavioral Generalization | 0,73926 | 0,74448 | 0,74738 | 0,77954 | 0,88510 |
| Behavioral Precision | 0,76113 | 0,76424 | 0,78264 | 0,99454 | 0,73878 |
| Behavioral Recall | 0,85507 | 0,93846 | 0,99501 | 0,96941 | 0,68383 |
| One Align | 0,99503 | 0,98730 | 0,99500 | 0,96509 | 0,80079 |
| Token Based Fitness | 0,88967 | 0,87550 | 0,94718 | 0,89188 | 0,70331 |

| | | | | | |
|---|---|---|---|---|---|
| Totale Communaliteit | 0,72384 | 0,75686 | 0,80553 | 0,91116 | KMO |
| Bartlett's p-waarde | 0,00000 | 0,00000 | 0,00000 | 0,00000 | 0,75838 |
| RMSR | 0,04037 | 0,03525 | 0,02967 | 0,00980 | |

Experiment 1b

| | Communaliteit | | | | |
|---|---|---|---|---|---|
| Metriek | 2 Factoren | 3 Factoren | 4 Factoren | 5 Factoren | MSA |
| Alignment Based Fitness | 0,14335 | 0,21662 | 0,63569 | 0,65861 | 0,48129 |
| Alignment Based Generalization | 0,00754 | 0,02510 | 0,12181 | 0,12789 | 0,50387 |
| Alignment Based Precision | 0,26812 | 0,66074 | 0,97327 | 0,64403 | 0,45884 |
| Best Align | 0,23820 | 0,26546 | 0,28920 | 0,50881 | 0,51203 |
| Behavioral Generalization | 0,34590 | 0,65985 | 0,56737 | 0,94708 | 0,64161 |
| Behavioral Precision | 0,71714 | 0,74385 | 0,75313 | 0,76807 | 0,54132 |
| Behavioral Recall | 0,99502 | 0,98129 | 0,97919 | 0,95319 | 0,56826 |
| One Align | 0,99501 | 0,99501 | 0,99501 | 0,99502 | 0,42527 |
| Token Based Fitness | 0,85564 | 0,88975 | 0,90503 | 0,95114 | 0,57598 |

| | | | | | |
|---|---|---|---|---|---|
| Totale Communaliteit | 0,50732 | 0,60419 | 0,69108 | 0,72820 | KMO |
| Bartlett's p-waarde | 0,00000 | 0,00000 | 0,00000 | 0,00000 | 0,52910 |
| RMSR | 0,09651 | 0,05223 | 0,02803 | 0,01212 | |

## Factorladingen voor experiment 1

| | 2-factoranalyse | | 3-factoranalyse | | |
|---|---|---|---|---|---|
| Metriek | Factor1 | Factor2 | Factor1 | Factor2 | Factor3 |
| Alignment Based Fitness | 0,86395 | 0,05353 | 0,71610 | 0,15918 | -0,32298 |
| Alignment Based Precision | -0,29321 | 0,65797 | -0,02260 | 0,33089 | 0,74911 |
| Best Align | 0,14584 | 0,88521 | 0,16045 | 0,78648 | 0,16267 |
| Behavioral Generalization | 0,65512 | -0,30823 | 0,69181 | -0,36503 | 0,08562 |
| Behavioral Precision | 0,11454 | 0,92180 | 0,04871 | 0,85682 | 0,04619 |
| Behavioral Recall | 0,95948 | 0,02207 | 1,02961 | -0,06639 | 0,14901 |
| Token Based Fitness | 0,99402 | 0,12026 | 0,91774 | 0,09957 | -0,07718 |
| Alignment Based | 0,02537 | 0,10145 | 0,01000 | 0,03687 | 0,08896 |
| One Align | -0,05613 | 0,96544 | -0,12203 | 0,93603 | 0,00375 |

## Factorladingen voor experiment 1a en 1b

Experiment 1a

| | 2-factoranalyse | | 3-factoranalyse | | |
|---|---|---|---|---|---|
| Metriek | Factor1 | Factor2 | Factor1 | Factor2 | Factor3 |
| Alignment Based Fitness | 0,0509 | 0,8855 | -0,0108 | 0,8132 | 0,0687 |
| Alignment Based | 0,1467 | 0,1094 | 0,0337 | -0,0184 | 0,3951 |
| Alignment Based Precision | 0,7241 | -0,2320 | 0,8048 | -0,1245 | -0,1392 |
| Best Align | 0,9253 | 0,1115 | 0,9672 | 0,1746 | -0,0556 |
| Behavioral Generalization | -0,3460 | 0,6224 | -0,3694 | 0,5897 | -0,0506 |
| Behavioral Precision | 0,9193 | 0,0927 | 0,8877 | 0,0717 | 0,2053 |
| Behavioral Recall | 0,0147 | 0,9346 | 0,0850 | 1,0429 | -0,2195 |
| One Align | 0,9534 | -0,0776 | 0,9650 | -0,0486 | 0,0790 |
| Token Based Fitness | 0,1205 | 1,0019 | 0,0534 | 0,9548 | 0,2681 |

| | 4-factoranalyse | | | |
|---|---|---|---|---|
| Metriek | Factor1 | Factor2 | Factor3 | Factor4 |
| Alignment Based Fitness | 0,0486 | 0,7699 | -0,2518 | -0,0462 |
| Alignment Based | 0,0075 | -0,0036 | 0,0734 | 0,4961 |
| Alignment Based Precision | 0,6567 | -0,0540 | 0,5595 | 0,0672 |
| Best Align | 0,9301 | 0,1604 | 0,1134 | -0,1161 |
| Behavioral Generalization | -0,4025 | 0,6211 | 0,1115 | 0,0850 |
| Behavioral Precision | 0,8635 | 0,0519 | 0,0539 | 0,1559 |
| Behavioral Recall | 0,0445 | 1,0564 | 0,1264 | -0,1719 |
| One Align | 0,9494 | -0,0759 | 0,0283 | -0,0097 |
| Token Based Fitness | 0,0645 | 0,9330 | -0,0798 | 0,2082 |

| | 5-factoranalyse | | | | |
|---|---|---|---|---|---|
| Metriek | Factor1 | Factor2 | Factor3 | Factor4 | Factor5 |
| Alignment Based Fitness | 0,0465 | 0,7990 | 0,0220 | -0,2467 | -0,0013 |
| Alignment Based | 0,0133 | -0,0062 | 0,9992 | 0,0323 | -0,0325 |
| Alignment Based Precision | 0,7161 | -0,0658 | 0,0476 | 0,5428 | 0,0316 |
| Best Align | 0,9796 | 0,1880 | -0,0117 | 0,0995 | -0,0511 |
| Behavioral Generalization | -0,3960 | 0,6189 | 0,0324 | 0,1143 | 0,0609 |
| Behavioral Precision | 0,8402 | 0,0047 | -0,0519 | 0,0363 | 0,5736 |
| Behavioral Recall | 0,0593 | 1,0538 | -0,1060 | 0,1336 | -0,1061 |
| One Align | 0,9198 | -0,0877 | 0,0359 | 0,0571 | 0,1277 |
| Token Based Fitness | 0,0711 | 0,9250 | 0,0894 | -0,0908 | 0,1008 |

Experiment 1b

| | 2-factoranalyse | | 3-factoranalyse | | |
|---|---|---|---|---|---|
| Metriek | Factor1 | Factor2 | Factor1 | Factor2 | Factor3 |
| Alignment Based Fitness | 0,3784 | -0,0052 | 0,1173 | 0,0398 | -0,3851 |
| Alignment Based | -0,4084 | 0,3109 | 0,1246 | 0,1857 | 0,8740 |
| Alignment Based Precision | 0,2134 | 0,4428 | 0,0548 | 0,4745 | -0,1736 |
| Best Align | 0,5851 | -0,0506 | 0,8230 | -0,1213 | 0,2376 |
| Behavioral Generalization | 0,1946 | 0,8278 | 0,0805 | 0,8506 | -0,0490 |
| Behavioral Precision | 0,9893 | -0,1110 | 0,7799 | -0,0924 | -0,4177 |
| Behavioral Recall | 0,9187 | 0,1276 | 0,8400 | 0,1262 | -0,2222 |
| One Align | 0,0869 | 0,0037 | 0,1529 | -0,0100 | 0,0652 |
| Token Based Fitness | -0,1856 | 0,9767 | -0,1158 | 0,9615 | 0,2923 |

| | 4-factoranalyse | | | |
|---|---|---|---|---|
| Metriek | Factor1 | Factor2 | Factor3 | Factor4 |
| Alignment Based Fitness | 0,1715 | -0,0026 | 0,0289 | 0,7629 |
| Alignment Based | -0,0657 | 0,1344 | 0,9790 | 0,0266 |
| Alignment Based Precision | 0,1188 | 0,4802 | -0,1746 | 0,0365 |
| Best Align | 0,7437 | -0,1140 | 0,2835 | -0,0728 |
| Behavioral Generalization | 0,1534 | 0,8485 | -0,0569 | -0,0402 |
| Behavioral Precision | 0,9147 | -0,1008 | -0,1272 | 0,1656 |
| Behavioral Recall | 0,9262 | 0,1295 | -0,0593 | -0,0077 |
| One Align | 0,1687 | 0,0085 | -0,0595 | -0,3352 |
| Token Based Fitness | -0,1599 | 0,9548 | 0,2089 | 0,0254 |

| | 5-factoranalyse | | | | |
|---|---|---|---|---|---|
| Metriek | Factor1 | Factor2 | Factor3 | Factor4 | Factor5 |
| Alignment Based Fitness | 0,1265 | 0,0109 | 0,0281 | -0,0070 | 0,7936 |
| Alignment Based | -0,0913 | 0,1630 | -0,0923 | 0,8118 | 0,0035 |
| Alignment Based Precision | -0,0690 | 0,0828 | 0,9714 | -0,0990 | 0,0249 |
| Best Align | 0,7351 | -0,1969 | 0,1355 | 0,3631 | -0,0358 |
| Behavioral Generalization | 0,2019 | 0,8545 | 0,0282 | -0,0860 | -0,0516 |
| Behavioral Precision | 0,8900 | -0,0568 | -0,0246 | -0,2136 | 0,1380 |
| Behavioral Recall | 0,9675 | 0,1954 | -0,1039 | -0,1041 | -0,0100 |
| One Align | 0,1633 | -0,0069 | 0,0430 | -0,0758 | -0,3337 |
| Token Based Fitness | -0,0986 | 0,9242 | 0,0454 | 0,2507 | 0,0475 |

# Bijlage B

## Data Preparation Experiment 1a

```
library(ggplot2)
library(cluster)
library(Hmisc)
library(plyr)
library(dplyr)
library(tidyr)
source("Scripts/functions.R")

## Inlezen data
data <- read.csv("Data/metrics_experiment_2.csv")

# pas datastructuur aan
data$completeness <- as.factor(data$completeness)
data$lognr <- as.factor(data$lognr)
data$noise <- as.factor(data$noise)

dataFinal <- data[, -1]

# verander de namen van de metrieken
levels(dataFinal$metric) <- c("Alignment Based Fitness",
  "Alignment Based Generalization", "Alignment Based Precision",
  "Best Align", "Behavioral Generalization", "Behavioral Precision",
  "Behavioral Recall", "One Align", "Token Based Fitness")

# voeg kolom dimensie toe
dataFinal$dimension <- "Precision"
dataFinal$dimension[dataFinal$metric == "Alignment Based Fitness" |
  dataFinal$metric == "Behavioral Recall" | dataFinal$metric ==
  "Token Based Fitness"] <- "Fitness"
dataFinal$dimension[dataFinal$metric == "Alignment Based Generalization" |
  dataFinal$metric == "Behavioral Generalization"] <- "Generalization"

# haal NA's en negatieve waarden uit de data
cleanedData <- dataFinal[!is.na(dataFinal$value) &
  dataFinal$value >= 0, ]


#### Pas datastructuur aan om factoranalyse over
#### metrieken mogelijk te maken
df <- cleanedData
df <- df[, -c(8, 9, 10)]
dfS <- spread(df, metric, value)
complete <- dfS[complete.cases(dfS), ]
# Er zijn veel observaties waar niet alle 10 metrieken correct berekend
# werden, en factoranalyse werkt niet met na's

# verzameling van niet complete cases
nc <- dfS[!complete.cases(dfS), ]
nc <- gather(nc, "Metriek", "Value", 6:14)
nc <- nc[is.na(nc$Value), ]
summary(nc)
missingValues <- as.data.frame(table(nc$Metriek))
# write.xlsx2(missingValues, 'Resulaten/Final/missingExp2.xlsx', row.names = F)


## Plot boxplot
boxplot <- plotBoxplot(dataFinal)
# ggsave('Resulaten/Final/boxplotExp2.png', boxplot)

cleanedData <- cleanedData[, c(-8, -9)]
rm(df, test)
```

### Data Preparation Experiment 1b

```
library(ggplot2)
library(cluster)
library(Hmisc)
library(plyr)
library(dplyr)
library(tidyr)
library(xlsx)
source("Scripts/functions.R")

## Inlezen data Experiment 1
data <- read.csv("Data/metrics_experiment_1.csv")
data2 <- read.csv("Data/metrics_experiment_1_sel.csv")
data3 <- read.csv("Data/metrics_experiment_1_Rozinat.csv")

# datastructuur aanpassen
data$X <- NULL
data$completeness <- as.factor(data$completeness)
data$lognr <- as.factor(data$lognr)
data$noise <- as.factor(data$noise)

data2$X <- NULL
data2$completeness <- as.factor(data2$completeness)
data2$lognr <- as.factor(data2$lognr)
data2$noise <- as.factor(data2$noise)
data2$log_id <- NULL

data3$X <- NULL
data3$completeness <- as.factor(data3$completeness)
data3$lognr <- as.factor(data3$lognr)
data3$noise <- as.factor(data3$noise)
data3$log_id <- NULL
data3$model_id <- NULL

data$dataset <- 1
data2$dataset <- 2
data3$dataset <- 3

# Voeg data samen en verwijder duplicate systemen
data <- data[data$metric != "RozinatFitness",]
data2 <- data2[data2$metric != "RozinatFitness",]
t <- rbind(data2, data, data3)
rownames(t) <- NULL
# duplicates <- t[duplicated(t[,c(-7, -8)]),]
t <- t[!duplicated(t[, c(-7, -8)]), ]  ## verwijder duplicaten

dataFinal <- t

# Verander de namen van de metrieken
levels(dataFinal$metric) <- c("Alignment Based Fitness",
  "Alignment Based Precision", "Best Align",
  "ETC Precision", "Behavioral Generalization",
  "Behavioral Precision", "Behavioral Recall",
  "Token Based Fitness", "Alignment Based Generalization",
  "One Align")
dataFinal <- dataFinal[dataFinal$metric != "ETC Precision",]
# niet opgenomen in analyse
dataFinal$metric <- factor(dataFinal$metric)

# Voeg kolom dimensie toe
dataFinal$dimension <- "Precision"
dataFinal$dimension[dataFinal$metric == "Alignment Based Fitness" |
  dataFinal$metric == "Behavioral Recall" |
  dataFinal$metric == "Token Based Fitness"] <- "Fitness"
```

```
dataFinal$dimension[dataFinal$metric == "Alignment Based Generalization" |
  dataFinal$metric == "Behavioral Generalization"] <- "Generalization"

# Verwijder data met negatieve waarde
cleanedData <- subset(dataFinal, dataFinal$value >= 0)
removedData <- subset(dataFinal, dataFinal$value < 0)

#### Pas datastructuur aan om factoranalyse over
#### metrieken mogelijk te maken
df <- cleanedData
df$dimension <- NULL
df$dataset <- NULL
row.names(df) <- NULL
dfS <- spread(df, metric, value)
complete <- dfS[complete.cases(dfS), ]
# Er zijn veel observaties waar niet alle 10
# metrieken correct berekend werden, en factor
# analyse werkt niet met na's

## Plot boxplot
boxplot <- plotBoxplot(cleanedData)
#ggsave("Resulaten/Final/boxplotExp1.png", boxplot)
rm(data, data2, data3, t, df)

# verzameling van niet complete cases
nc <- dfS[!complete.cases(dfS), ]
nc <- gather(nc, "Metriek", "Value", 6:14)
nc <- nc[is.na(nc$Value), ]
summary(nc)
missingValues <- as.data.frame(table(nc$Metriek))
#write.xlsx2(missingValues, "Resulaten/Final/missingExp1.xlsx",
#row.names = F)
```

### Voeg data experiment 1a en 1b samen

```
# Exp 1
source("Scripts/dataPrep.R")
exp1 <- cleanedData
exp1$experiment <- "1"
# verwijder tabellen die niet nodig zijn
rm(cleanedData, dataFinal, dfS, removedData, complete,
  boxplot, missingValues, nc)

# Exp 2
source("Scripts/dataPrepExp2.R")
exp2 <- cleanedData
exp2$experiment <- "2"
# verwijder tabellen die niet nodig zijn
rm(cleanedData, complete, data, dataFinal, missingValues,
  nc, boxplot)

# voeg samen
data <- rbind(exp1[, c(1:7, 10)], exp2[, c(1:7, 9)])
rm(exp1, exp2)

# Voeg dimensie toe
d <- data
d$dimension <- "Precision"
d$dimension[d$metric == "Alignment Based Fitness" |
  d$metric == "Behavioral Recall" | d$metric == "Token Based Fitness"] <- "Fitness"
d$dimension[d$metric == "Alignment Based Generalization" |
```

```
  d$metric == "Behavioral Generalization"] <- "Generalization"

# maak boxplot
boxplotTotal <- plotBoxplot(d)
# ggsave('Resulaten/Final/boxplotTotal.png',boxplotTotal)

# verander structuur om correlatiematrix op te stellen
row.names(d) <- NULL
temp <- spread(d[, -9], metric, value)
dFa <- temp[, c(7:15)]  # behoud enkel de waarden van de metrieken
dFa <- dFa[complete.cases(dFa), ]
corMatTot <- plotCorrelationMatrix(dFa)
# ggsave('Resulaten/Final/correlatieMatrixExp1Tot.png', corMatTot)
rm(d, dFa, temp, dfS)
```

## Analyse ontbrekende waarden

```
source("Scripts/dataPrep.R")
test <- gather(dfS, "metric", "value", 6:14)

temp <- test
c <- as.data.frame(table(temp$metric[!complete.cases(temp$value)]))

nv <- subset(dataFinal, dataFinal$value < 0)
ncomplete <- subset(dataFinal, !complete.cases(dataFinal))

# histogram van ontbrekende waarden per dimensie
histMissingValues <- ggplot(ncomplete, aes(metric)) +
  geom_bar(stat = "count", aes(fill = dimension)) +
  ggtitle("Ontbrekende waarden") + theme_bw_angle() +
  xlab("Metriek") + ylab("Aantal") + geom_text(stat = "count",
  aes(label = ..count.., y = (..count..) + 15)) +
  scale_fill_discrete(name = "Kwaliteits \nDimensie")
histMissingValues
ggsave("histMissingValues.png", histMissingValues)


source("Scripts/dataPrepExp2.R")
test <- gather(dfS, "metric", "value", 6:14)
temp <- test
c <- as.data.frame(table(temp$metric[!complete.cases(temp$value)]))
```

## Factoranalyse experiment 1

```
source("Scripts/mergeDatasets.R")
library(psych)
library(nFactors)
library(xlsx)
library(psy)
source("Scripts/qualityFactorAnalysis.R")

# Factoranalyse
# bepaal factoren
row.names(data) <- NULL
dFa <- spread(data[,-9], metric, value)
dFa <- dFa[,c(7:15)] # behoud enkel de waarden van de metrieken
dFa <- dFa[complete.cases(dFa),]
corMatrix <- cor(dFa) # stel correlatiematrix op
fit <- factanal(dFa, 5)
# De keuze van het aantal factoren heeft geen invloed op de
# opstelling van een scree plot
```

```
scree.plot(fit$correlation) # maak scree plot


## 2 factoren
fit2 <- factanal(dFa, 2, rotation = "promax")
plot2Factors <- plotFactorLoadings(fit2, 2)
ggsave("Resulaten/Final/TweeFactorAnalyseExp1.png", plot2Factors)
quality2 <- qualityCheckFA(dFa, 2)
write.xlsx2(x = quality2, file="Resulaten/qualityFA_2_Exp1_Total.xlsx")

## 3 factoren
fit3 <- factanal(dFa, 3, rotation = "promax")
plot3Factors <- plotFactorLoadings(fit3, 3)
ggsave("Resulaten/Final/DrieFactorAnalyseExp1.png", plot3Factors)
quality3 <- qualityCheckFA(dFa, 3)
write.xlsx2(x = quality3, file="Resulaten/qualityFA_3_Exp1_Total.xlsx")


# haal alle factorladingen op
exp1 <- list()
exp1$two <- as.table(fit2$loadings)
exp1$three <- as.table(fit3$loadings)
# schrijf weg in excel file
for(i in 1:2){
  name <- paste("Resulaten/Final/loadingsExp1_Tot_", i+1, ".xlsx", sep = "")
  df <- as.data.frame(exp1[i])
  names(df) <- c("var1", "var2", "var3")
  write.xlsx2(spread(df, var2, var3), name)
}
```

### Gevoeligheid aan kwaliteit experiment 1

```
source("Scripts/mergeDatasets.R")
row.names(data) <- NULL
data_spread <- spread(data, metric, value)
data_spread <- data_spread[, 7:15]
data_spread <- data_spread[complete.cases(data_spread),]

se <- T  # met of zonder standaard afwijking?
method <- "loess"  # methode van smoothing

# Plot voor fitnessmetrieken
f <- ggplot(data_spread, aes(x = `Token Based Fitness`)) +
  stat_smooth(aes(y = `Token Based Fitness`, colour = "Token Based Fitness"),
    se = se, method = method) + stat_smooth(aes(y = `Behavioral Recall`,
  colour = "Behavioral Recall"), se = se, method = method) +
  stat_smooth(aes(y = `Alignment Based Fitness`,
    colour = "Alignment Based Fitness"), se = se,
    method = method) + coord_fixed(ratio = 1) +
  coord_cartesian(xlim = c(0, 1)) + theme_bw() +
  scale_colour_discrete("Metriek") + guides(
    color = guide_legend(override.aes = list(fill = NA))) +
  ylab("Token Based Fitness\nBehavioral Recall")
ggsave("Resulaten/Final/complexityFitness.png", f)

# Plot voor precisionmetrieken
p <- ggplot(data_spread, aes(x = `One Align`)) + stat_smooth(
  aes(y = `Alignment Based Precision`,
  colour = "Alignment Based Precision"), se = se,
  method = method) + stat_smooth(aes(y = `Best Align`,
  colour = "Best Align"), se = se, method = method) +
  stat_smooth(aes(y = `Behavioral Precision`, colour = "Behavioral Precision"),
    se = se, method = method) + stat_smooth(aes(y = `One Align`,
```

```
  colour = "One Align"), se = se, method = method) +
  coord_fixed(ratio = 1) + coord_cartesian(xlim = c(0,
  1)) + scale_colour_discrete("Metriek") + guides(color = guide_legend(
    override.aes = list(fill = NA))) +
  ylab("Alignment Based Precision   Best Align\nBehavioral Precision   One Align") +
  theme_bw()
ggsave("Resulaten/Final/complexityPrecision.png", p)
```

### Factoranalyse experiment 1a

```
source("Scripts/dataPrep.R")
library(psych)
library(nFactors)
library(xlsx)
library(psy)
source("Scripts/qualityFactorAnalysis.R")

# Factoranalyse
# bepaal factoren
dFa <- complete[,c(6:14)] # behoud enkel de waarden van de metrieken
corMatrix <- cor(dfS[,c(6:14)]) # stel correlatiematrix op
fit <- factanal(dFa, 5)
# De keuze van het aantal factoren heeft geen invloed op de
# opstelling van een scree plot
scree.plot(fit$correlation) # maak scree plot

ggsave("Resulaten/Final/correlatieMatrixExp1a.png", plotCorrelationMatrix(dFa))

## 2 factoren
fit2 <- factanal(dFa, 2, rotation = "promax")
plot2Factors <- plotFactorLoadings(fit2, 2)
ggsave("Resulaten/Final/TweeFactorAnalyseExp1.png", plot2Factors)
quality2 <- qualityCheckFA(dFa, 2)
write.xlsx2(x = quality2, file="Resulaten/qualityFA_2_Exp1.xlsx")

## 3 factoren
fit3 <- factanal(dFa, 3, rotation = "promax")
plot3Factors <- plotFactorLoadings(fit3, 3)
ggsave("Resulaten/Final/DrieFactorAnalyseExp1.png", plot3Factors)
quality3 <- qualityCheckFA(dFa, 3)
write.xlsx2(x = quality3, file="Resulaten/qualityFA_3_Exp1.xlsx")

## 4 factoren
fit4 <- factanal(dFa, 4, rotation ="promax")
plot4Factors <- plotFactorLoadings(fit4,4)
ggsave("Resulaten/Final/VierFactorAnalysePromaxExp1.png", plot4Factors)
quality4 <- qualityCheckFA(dFa, 4)
write.xlsx2(x = quality4, file="Resulaten/qualityFA_4_Exp1.xlsx")

## 5 factoren
fit5 <- factanal(dFa, 5, rotation="promax")
plot5Factors <- plotFactorLoadings(fit5,5)
ggsave("Resulaten/Final/VijfFactorAnalysePromaxExp1.png", plot5Factors)
quality5 <- qualityCheckFA(dFa, 5)
write.xlsx2(x = quality5, file="Resulaten/qualityFA_5_Exp1.xlsx")

# haal alle factorladingen op
exp1 <- list()
exp1$two <- as.table(fit2$loadings)
exp1$three <- as.table(fit3$loadings)
exp1$four <- as.table(fit4$loadings)
exp1$five <- as.table(fit5$loadings)
# schrijf de ladingen weg in excel file
```

```
for(i in 1:4){
  name <- paste("Resulaten/Final/loadingsExp1_", i+1, ".xlsx", sep = "")
  df <- as.data.frame(exp1[i])
  names(df) <- c("var1", "var2", "var3")
  write.xlsx2(spread(df, var2, var3), name)
}
```

### Factoranalyse experiment 1b

```
source("Scripts/dataPrepExp2.R")
library(psych)
library(nFactors)
library(plyr)
library(xlsx)
source("Scripts/qualityFactorAnalysis.R")

## Bepaal factoren
dFa <- complete[,c(6:14)] # behoud enkel de waarden van de metrieken

corMatrix <- cor(dFa) # stel correlatiematrix op
fit <- factanal(dFa, 5)
scree.plot(fit$correlation) # De keuze van het aantal factoren heeft geen invloed voor het
                            # opstellen van de screeplot

ggsave("Resulaten/Final/correlatieMatrixExp2.png", plotCorrelationMatrix(dFa))

## 2 factoren
fit2 <- factanal(dFa, 2, rotation = "promax")
plot2Factors <- plotFactorLoadings(fit2, 2)
ggsave("Resulaten/Final/TweeFactorAnalyseExp2.png", plot2Factors)
quality2 <- qualityCheckFA(dFa, 2)
write.xlsx2(x = quality2, file="Resulaten/qualityFA_2.xlsx")

## 3 factoren
fit3 <- factanal(dFa, 3, rotation = "promax")
plot3Factors <- plotFactorLoadings(fit3, 3)
ggsave("Resulaten/Final/DrieFactorAnalyseExp2.png", plot3Factors)
quality3 <- qualityCheckFA(dFa, 3)
write.xlsx2(x = quality3, file="Resulaten/qualityFA_3.xlsx")

## 4 factoren
fit4 <- factanal(dFa, 4, rotation ="promax")
plot4Factors <- plotFactorLoadings(fit4,4)
ggsave("Resulaten/Final/VierFactorAnalysePromaxExp2.png", plot4Factors)
quality4 <- qualityCheckFA(dFa, 4)
write.xlsx2(x = quality4, file="Resulaten/qualityFA_4.xlsx")

## 5 factoren
fit5 <- factanal(dFa, 5, rotation="promax")
plot5Factors <- plotFactorLoadings(fit5,5)
ggsave("Resulaten/Final/VijfFactorAnalysePromaxExp2.png", plot5Factors)
quality5 <- qualityCheckFA(dFa, 5)
write.xlsx2(x = quality5, file="Resulaten/qualityFA_5.xlsx")

# haal alle factorladingen op
exp2 <- list()
exp2$two <- as.table(fit2$loadings)
exp2$three <- as.table(fit3$loadings)
exp2$four <- as.table(fit4$loadings)
exp2$five <- as.table(fit5$loadings)
for(i in 1:4){
  name <- paste("Resulaten/Final/loadingsExp2_", i+1, ".xlsx", sep = "")
  df <- as.data.frame(exp2[i])
```

```
  names(df) <- c("var1", "var2", "var3")
  write.xlsx2(spread(df, var2, var3), name)
}
```

### Data Preparation experiment 2

```
library(tidyr)
library(ggplot2)
library(Rmisc)
source("Scripts/functions.R")

# data_wide <-
# read.csv2('Data/metrics_experiment_4_wide.csv', sep=',')
data_long <- read.csv2("Data/metrics_experiment_4_long(2).csv",
  sep = ",")

data_long$completeness <- as.factor(data_long$completeness)
data_long$lognr <- as.factor(data_long$lognr)
data_long$noise <- as.factor(data_long$noise)
data_long$value <- as.numeric(as.character(data_long$value))

# verander namen van de metrieken
levels(data_long$metric) <- c("Alignment Based Fitness",
  "Alignment Based Generalization", "Alignment Based Precision",
  "Best Align", "Behavioral Generalization", "Behavioral Precision",
  "Behavioral Recall", "One Align", "Token Based Fitness")

# voeg dimensie toe
data_long$dimension <- "Precision"
data_long$dimension[data_long$metric == "Alignment Based Fitness" |
  data_long$metric == "Behavioral Recall" | data_long$metric ==
  "Token Based Fitness"] <- "Fitness"
data_long$dimension[data_long$metric == "Alignment Based Generalization" |
  data_long$metric == "Behavioral Generalization"] <- "Generalization"

# plot een boxplot en save
# ggsave('Resulaten/Final/boxplotExp3.png',plotBoxplot(data_long))


# bereken gemiddelde per metriek, per noise-level
avgData <- summarySE(data_long[-1], measurevar = "value",
  groupvars = c("metric", "noise"))
levels(avgData$noise) <- c("0", "5", "10", "15", "20",
  "25", "30", "35", "40", "45")
# voeg opnieuw dimensie toe
avgData$dimension <- "Precision"
avgData$dimension[avgData$metric == "Alignment Based Fitness" |
  avgData$metric == "Behavioral Recall" | avgData$metric ==
  "Token Based Fitness"] <- "Fitness"
avgData$dimension[avgData$metric == "Alignment Based Generalization" |
  avgData$metric == "Behavioral Generalization"] <- "Generalization"

# plot gemiddelde per metriek, per noise level
pd <- position_dodge(0.1)
influenceNoise <- ggplot(avgData, aes(x = noise, y = value,
  colour = metric, group = metric)) + geom_errorbar(aes(ymin = value -
  ci, ymax = value + ci), colour = "black", width = 0.1,
  position = pd) + geom_line(position = pd) + geom_point(position = pd,
  size = 3) + facet_wrap(~dimension, nrow = 3) +
  xlab("Non-fitting cases (%)") + theme_bw()
# ggsave('Resulaten/Final/influenceNoise.png',influenceNoise)

# bereken gemiddelde per metriek, per noise-level,
```

```
# per systeem
avgDataPerSystem <- summarySE(data_long[-1], measurevar = "value",
  groupvars = c("metric", "noise", "system"))
levels(avgDataPerSystem$noise) <- c("0", "5", "10",
  "15", "20", "25", "30", "35", "40", "45")
# voeg opnieuw dimensie toe
avgDataPerSystem$dimension <- "Precision"
avgDataPerSystem$dimension[avgDataPerSystem$metric ==
  "Alignment Based Fitness" | avgDataPerSystem$metric ==
  "Behavioral Recall" | avgDataPerSystem$metric ==
  "Token Based Fitness"] <- "Fitness"
avgDataPerSystem$dimension[avgDataPerSystem$metric ==
  "Alignment Based Generalization" | avgDataPerSystem$metric ==
  "Behavioral Generalization"] <- "Generalization"

# plot gemiddelde per metriek, per noise level, per
# systeem
pd <- position_dodge(0.1)
influenceNoiseSystem <- ggplot(avgDataPerSystem, aes(x = noise,
  y = value, colour = metric, group = metric)) +
  geom_errorbar(aes(ymin = value - ci, ymax = value +
    ci), colour = "black", width = 0.1, position = pd) +
  geom_line(position = pd) + geom_point(position = pd,
  size = 2) + facet_grid(system ~ dimension) + xlab("Non-fitting cases (%)") +
  theme_bw()
# ggsave('Resulaten/Final/influenceNoiseSystem.png',influenceNoiseSystem)

# bereken gemiddelde per metriek, per noise-level,
# per compl niveau
avgDataCompl <- summarySE(data_long[-1], measurevar = "value",
  groupvars = c("metric", "noise", "completeness"))
levels(avgDataPerSystem$noise) <- c("0", "5", "10",
  "15", "20", "25", "30", "35", "40", "45")
# voeg opnieuw dimensie toe
avgDataCompl$dimension <- "Precision"
avgDataCompl$dimension[avgDataCompl$metric == "Alignment Based Fitness" |
  avgDataCompl$metric == "Behavioral Recall" | avgDataCompl$metric ==
  "Token Based Fitness"] <- "Fitness"
avgDataCompl$dimension[avgDataCompl$metric == "Alignment Based Generalization" |
  avgDataCompl$metric == "Behavioral Generalization"] <- "Generalization"

# plot gemiddelde per metriek, per noise level
pd <- position_dodge(0.1)
influenceNoiseCompl <- ggplot(avgDataCompl, aes(x = noise,
  y = value, colour = metric, group = metric)) +
  geom_errorbar(aes(ymin = value - ci, ymax = value +
    ci), colour = "black", width = 0.1, position = pd) +
  geom_line(position = pd) + geom_point(position = pd,
  size = 3) + facet_grid(dimension ~ completeness) +
  theme_bw()
# ggsave('Resulaten/Final/influenceNoiseCompl.png',
# influenceNoiseCompl)

## missing values
data_spread_all <- spread(data_long[, -c(1, 8, 9)],
  metric, value)
mv <- gather(data_spread_all[!complete.cases(data_spread_all),
  ], "metric", "value", 5:13)
mv$meanFitness <- NULL
mv <- mv[!complete.cases(mv), ]
levels(mv$noise) <- c("0", "5", "10", "15", "20", "25",
  "30", "35", "40", "45")
# voeg opnieuw dimensie toe
mv$dimension <- "Precision"
mv$dimension[mv$metric == "Alignment Based Fitness" |
  mv$metric == "Behavioral Recall" | mv$metric ==
```

```
  "Token Based Fitness"] <- "Fitness"
mv$dimension[mv$metric == "Alignment Based Generalization" |
  mv$metric == "Behavioral Generalization"] <- "Generalization"

# verander volgorde voor de plot
mv$metric <- factor(mv$metric, levels = c("One Align",
  "Best Align", "Alignment Based Precision", "Behavioral Precision",
  "Alignment Based Fitness", "Behavioral Recall",
  "Token Based Fitness", "Behavioral Generalization",
  "Alignment Based Generalization"))

# plot ontbrekende waarden
missingValues <- ggplot(mv, aes(x = noise)) + geom_bar(stat = "count",
  aes(fill = dimension)) + facet_wrap(~metric) +
  theme_bw() + xlab("Non-fitting cases (%)") + ylab("Aantal")
# ggsave('Resulaten/Final/missingValues.png',missingValues)

# verzamel data die gebruik wordt voor de anova test
temp <- gather(data_spread_all, "metric", "value",
  5:13)
# write.csv2(temp, 'test.csv')

# voer anova test uit
t <- anova(lm(value ~ noise * metric, temp))
# metriek en noise hebben significante interactie
# write.csv2(t, 'Resulaten/Final/anovaExp2.csv') #
# schrijf weg naar csv

# aparte anova per metriek (plus wegschrijven in excel)
for (i in unique(temp$metric)) {
  t <- anova(lm(value ~ noise, subset(temp, metric ==
    i)))
  path <- paste("Resulaten/Final/", i, "Anova.csv")
  write.csv2(t, path)
  print(i)
  print(t)
  cat("\n", "--------------------------------------------------",
    "\n")
}

attach(temp)
qqnorm(value)
qqline(value)
hist(value)
# niet normaal verdeeld!! -> andere niet-parametrische test uitvoeren

# voer kruskal wallis test uit
for (i in 5:13) {
  name <- names(data_spread_all)[i]
  test <- kruskal.test(data_spread_all[, i] ~ noise,
    data = data_spread_all)
  print(name)
  print(test)
  cat("\n\n", "--------------------", "\n")
}
```

## Regressieanalyse experiment 2

```
source("Scripts/DataPrepExp3.R")
# verander levels van noise naar % non-fitting cases
data_spread_all$noise <- revalue(data_spread_all$noise,
  c(`0` = "0", `1` = "0.05", `2` = "0.10", `3` = "0.15",
    `4` = "0.20", `5` = "0.25", `6` = "0.30", `7` = "0.35",
    `8` = "0.40", `9` = "0.45"))
data_spread_all$noise <- as.numeric(as.character(data_spread_all$noise))

# lineaire regressie met noise als numerische waarde Precision
summary(lm1 <- lm(`One Align` ~ noise + completeness +
                    system, data = data_spread_all))
summary(lm4 <- lm(`Behavioral Precision` ~ noise +
                    completeness + system, data = data_spread_all))

# Fitness
summary(lm1F <- lm(`Token Based Fitness` ~ noise +
                     completeness + system, data = data_spread_all))
summary(lm2F <- lm(`Alignment Based Fitness` ~ noise +
                     completeness + system, data = data_spread_all))
summary(lm3F <- lm(`Behavioral Recall` ~ noise +
                     completeness + system, data = data_spread_all))

# Generalization
summary(lm1G <- lm(`Alignment Based Generalization` ~ noise +
                     completeness + system, data = data_spread_all))

summary(lm2G <- lm(`Behavioral Generalization` ~ noise + I(noise^2) +
completeness + system, data = data_spread_all))


# haal coëfficiënten, R-squared en residual error op en voeg samen in dataframe
OneAlign <- lm1$coefficients
OneAlign$r.squared <- summary(lm1)$r.squared
OneAlign$sigma <- summary(lm1)$sigma

BehavioralPrecision <- lm4$coefficients
BehavioralPrecision$r.squared <- summary(lm4)$r.squared
BehavioralPrecision$sigma <- summary(lm4)$sigma

TokenBasedFitness <- lm1F$coefficients
TokenBasedFitness$r.squared <- summary(lm1F)$r.squared
TokenBasedFitness$sigma <- summary(lm1F)$sigma

AlignmentBasedFitness <- lm2F$coefficients
AlignmentBasedFitness$r.squared <- summary(lm2F)$r.squared
AlignmentBasedFitness$sigma <- summary(lm2F)$sigma

BehavioralRecall <- lm3F$coefficients
BehavioralRecall$r.squared <- summary(lm3F)$r.squared
BehavioralRecall$sigma <- summary(lm3F)$sigma

AlignmentBasedGeneralization <- lm1G$coefficients
AlignmentBasedGeneralization$r.squared <- summary(lm1G)$r.squared
AlignmentBasedGeneralization$sigma <- summary(lm1G)$sigma

BehavioralGeneralization <- lm2G$coefficients
BehavioralGeneralization$r.squared <- summary(lm2G)$r.squared
BehavioralGeneralization$sigma <- summary(lm2G)$sigma

# voeg samen
coefficients <- rbind(OneAlign, BehavioralPrecision,
                      TokenBasedFitness, AlignmentBasedFitness,
                      BehavioralRecall, AlignmentBasedGeneralization)
```

```r
# schrijf coefficienten weg naar excel file
library(xlsx)
write.xlsx2(t(coefficients), "Resulaten/Final/coefficientenMetComplEnSystem.xlsx")
write.xlsx2(BehavioralGeneralization, "Resulaten/Final/coefficientsBehGen.xlsx")
```

### Gebruikte functies

```r
library(tidyr)
library(grid) #for adjusting plot margins
library(gridExtra)
library(reshape2)

# thema voor de plots (b&w en schuine labels voor x-as)
theme_bw_angle <- function(base_size = 12){
  theme_bw(base_size = base_size) %+replace%
    theme(
      axis.title = element_text(size = 12),
      axis.text.x = element_text(angle = 30, hjust = 1, vjust = 0.95)
    )
}


# functie om factorladingen te plotten
plotFactorLoadings <- function(fit, i){
  loadings <- as.data.frame(fit$loadings[,1:i])
  loadings$metric <- row.names(loadings)

  loadings <- gather(loadings, "Factor", "Loading", 1:i)
  loadings$dimension <- "Precision"
  loadings$dimension[loadings$metric == "Alignment Based Fitness" |
                       loadings$metric == "Behavioral Recall" | loadings$metric ==
                       "Token Based Fitness"] <- "Fitness"
  loadings$dimension[loadings$metric == "Alignment Based Generalization" |
                       loadings$metric == "Behavioral Generalization"] <- "Generalization"

  plotLoadings <- ggplot(loadings, aes(metric, Loading)) +
    geom_bar(stat="identity", aes(fill=dimension)) +
    facet_wrap(~Factor, ncol=i) +
    theme_bw() +
    xlab("Metriek") +
    ylab("Lading") +
    geom_hline(aes(yintercept = 0), colour="darkgrey") +
    coord_flip() +
    scale_fill_discrete(name="Kwaliteits\nDimensie")
  plotLoadings
}

# functie om boxplots op te stellen
plotBoxplot <- function(data){
  temp <- data
  temp$metric <- factor(temp$metric, levels = temp$metric[order(temp$dimension)]) #order

  indx <- sapply(temp, is.factor)
  temp[indx] <- lapply(temp[indx], function(x) {
    levels(x) <- make.unique(levels(x))
    x })

  temp <- temp[temp$value>=0,] # verwijder -1 voor AryaFitness
  p <- ggplot(temp, aes(metric, value)) +
    geom_boxplot(aes(fill=dimension)) +
    xlab("Metriek") +
    ylab("Waarde") +
```

```
    scale_fill_discrete(name="Kwaliteits\ndimensie") +
    theme_bw_angle() +
    ggtitle("Boxplot") +
    theme(plot.margin = unit(c(0.1,0.1,0.1,1.5), "cm")) +
    stat_summary(fun.y=mean, colour="darkred", geom="point", size=3) # voeg gemiddelde toe
  p
}

# functie om correlatiematrix op te stellen
plotCorrelationMatrix <- function(data){
  ## Visualisatie Factoren http://rpubs.com/danmirman/plotting_factor_analysis
  corMatrix <- cor(data)
  corrs.m <- melt(corMatrix, id="Test", variable.name="Test2", value.name = "Correlation")

  corrs.m$dimension1 <- "Precision"
  corrs.m$dimension1[corrs.m$Var1 == "Alignment Based Fitness" | corrs.m$Var1 == "Behaviora
l Recall" |
                       corrs.m$Var1 == "Token Based Fitness"] <- "Fitness"
  corrs.m$dimension1[corrs.m$Var1 == "Alignment Based Generalization" | corrs.m$Var1 ==
                       "Behavioral Generalization"] <- "Generalization"

  corrs.m$dimension2 <- "Precision"
  corrs.m$dimension2[corrs.m$Var2 == "Alignment Based Fitness" | corrs.m$Var2 == "Behaviora
l Recall" |
                       corrs.m$Var2 == "Token Based Fitness"] <- "Fitness"
  corrs.m$dimension2[corrs.m$Var2 == "Alignment Based Generalization" | corrs.m$Var2 ==
                       "Behavioral Generalization"] <- "Generalization"

  #Sorteren volgens dimensie
  corrs.m$Var1 <-factor(corrs.m$Var1, levels=corrs.m[order(corrs.m$dimension1), "Var1"])
  corrs.m$Var2 <- factor(corrs.m$Var2, levels=corrs.m[order(corrs.m$dimension2), "Var2"])

  #place the tests on the x- and y-axes,
  #fill the elements with the strength of the correlation
  p1 <- ggplot(corrs.m, aes(Var1, Var2, fill=abs(Correlation))) +
    geom_tile() + #rectangles for each correlation
    #add actual correlation value in the rectangle
    geom_text(aes(label = round(Correlation, 2)), size=3.5) +
    theme_bw(base_size=14) + #black and white theme with set font size
    #rotate x-axis labels so they don't overlap,
    #get rid of unnecessary axis titles
    #adjust plot margins
    theme(axis.text.x = element_text(angle = 90),
          axis.title.x=element_blank(),
          axis.title.y=element_blank()
          #plot.margin = unit(c(0, 0, 0, 5), "cm")
    ) +
    #set correlation fill gradient
    scale_fill_gradient(low="white", high="red") +
    guides(fill=F) #omit unnecessary gradient legend
  p1
}


# functie om RMSR te berekenen uit de errors
rmse <- function(error) {sqrt(mean(error^2))}

# functie om coëfficienten uit te printen
printCoeff <- function(model){
  VARIABLE=c("",gsub("[-^0-9]", "", names(unlist(model$xlevels))))
  MODALITY=c("",as.character(unlist(model$xlevels)))
  names=data.frame(VARIABLE,MODALITY,NOMVAR=c("(Intercept)",paste(VARIABLE,MODALITY,sep=""))
[-1]))
  regression=data.frame(NOMVAR=names(coefficients(model)),COEF=as.numeric(coefficients(mode
l)))
```

```
  t<-merge(names,regression)
  names(t) <- c("name","variable","modality","coefficient")
  return(t)
}
```

### Functie voor kwaliteitsanalyse factoranalyse

```
# Kwaliteitscheck factoranalyse
library(psych)

## Functie voor kwaliteitscheck factoranalyse
qualityCheckFA <- function(dataset, factors){
  require(psych)
  corMat <- cor(dataset)
  solution <- fa(r=corMat, nfactors = factors, residuals = T)
  singleComm <- solution$communality
  singleComm <- as.data.frame(singleComm)
  names(singleComm) <- "Communality"

  ## Kaiser-Meyer-Olkin Measure of Sampling
  kmo <- KMO(corMat)

  ## Bartlett's test of Sphericity
  n <- length(names(dataset))
  degreeOfFreedom <- (n*n - n) / 2
  bartlett <- cortest.bartlett(corMat, degreeOfFreedom)

  ## RMSR
  rmsr <- solution$rms

  ## Total Communality
  tComm <- sum(singleComm)/n

  # Individuele MSA
  indMSA <- as.data.frame(kmo$MSAi)

  cat("Bartlett p-value: ", bartlett$p.value, "\nKMO: ", kmo$MSA, "\nRMSR: ", rmsr)
  cat("\nCommunalities:")
  cat("\nTotal Communality: ", tComm)

  quality <- cbind(singleComm, indMSA)
  return(quality)
}
```

# Auteursrechtelijke overeenkomst

Ik/wij verlenen het wereldwijde auteursrecht voor de ingediende eindverhandeling:
**Een analyse van de evaluatiemetrieken voor process discovery**

Richting: **master in de toegepaste economische wetenschappen: handelsingenieur in de beleidsinformatica**
Jaar: **2016**

in alle mogelijke mediaformaten, - bestaande en in de toekomst te ontwikkelen - , aan de Universiteit Hasselt.

Niet tegenstaand deze toekenning van het auteursrecht aan de Universiteit Hasselt behoud ik als auteur het recht om de eindverhandeling, - in zijn geheel of gedeeltelijk -, vrij te reproduceren, (her)publiceren of distribueren zonder de toelating te moeten verkrijgen van de Universiteit Hasselt.

Ik bevestig dat de eindverhandeling mijn origineel werk is, en dat ik het recht heb om de rechten te verlenen die in deze overeenkomst worden beschreven. Ik verklaar tevens dat de eindverhandeling, naar mijn weten, het auteursrecht van anderen niet overtreedt.

Ik verklaar tevens dat ik voor het materiaal in de eindverhandeling dat beschermd wordt door het auteursrecht, de nodige toelatingen heb verkregen zodat ik deze ook aan de Universiteit Hasselt kan overdragen en dat dit duidelijk in de tekst en inhoud van de eindverhandeling werd genotificeerd.

Universiteit Hasselt zal mij als auteur(s) van de eindverhandeling identificeren en zal geen wijzigingen aanbrengen aan de eindverhandeling, uitgezonderd deze toegelaten door deze overeenkomst.

Voor akkoord,

**Donders, Niels**

Datum: **28/05/2016**