UHASSELT

KNOWLEDGE IN ACTION

Faculteit Bedrijfseconomische Wetenschappen

master handelsingenieur in de beleidsinformatica

Masterthesis

Het analyseren van complexe processen: een vergelijkende analyse van verschillende trace clustering technieken

Joris Ganne

Scriptie ingediend tot het behalen van de graad van master handelsingenieur in de beleidsinformatica

PROMOTOR :

dr. Gert JANSSENSWILLEN

2021

2022

UHASSELT

KNOWLEDGE IN ACTION

www.uhasselt.be
Universiteit Hasselt
Campus Hasselt:
Martelarenlaan 42 | 3500 Hasselt
Campus Diepenbeek:
Agoralaan Gebouw D | 3590 Diepenbeek

UHASSELT

KNOWLEDGE IN ACTION

Faculteit Bedrijfseconomische Wetenschappen

master handelsingenieur in de beleidsinformatica

Masterthesis

Het analyseren van complexe processen: een vergelijkende analyse van verschillende trace clustering technieken

Joris Ganne

Scriptie ingediend tot het behalen van de graad van master handelsingenieur in de beleidsinformatica

PROMOTOR :

dr. Gert JANSSENSWILLEN

Voorwoord

In het academiejaar 2017-2018 begon ik aan de opleiding Handelsingenieur aan de Universiteit Hasselt. Ik koos na lang twijfelen,ik twijfelde tussen handelsingenieur en industrieel ingenieur, voor deze opleiding omdat een handelsingenieur een brede blik heeft op de maatschappij en dat paste goed bij de persoon die ik toen was en nog altijd ben: een persoon met vele interesses. Doorheen het eerste jaar besefte ik al snel dat ik in het tweede jaar zou kiezen voor Handelsingenieur in de beleidsinformatica. De combinatie van bedrijfseconomische inzichten en deze toepassen met behulp van IT en data, en dit op een proces-gericht manier, was voor mij persoonlijk het ideale evenwicht. Doorheen de bachelor- en masteropleiding gaf de opleiding aan de Universiteit Hasselt mij de kans om mij te verdiepen in de combinatie van bedrijfsprocessen en data en hoe hier samen waarde uit te halen. Deze thesis liet me toe om me verder te verdiepen in deze combinatie en zo een sluitstuk te vormen van deze opleiding. Deze thesis is een werk van lange adem en was slechts mogelijk door de steun van een heel aantal personen.

Allereerst wens ik mijn promotor dr. Gert Janssenswillen te bedanken voor zijn ondersteuning door het hele proces en voor zijn constructieve en waardevolle feedback. Daarnaast wil ik ook de docenten van de vakgroep beleidsinformatica bedanken om de passie voor data in mij aan te wakkeren en levende te houden. Verder wil ik ook zeker mijn ouders bedanken die mij doorheen de hele periode aan de UHasselt, en ver daarvoor al, gesteund hebben en ervoor gezorgd hebben dat ik geprikkeld werd door tal van domeinen. Tot slot bedank ik ook graag mijn vriendin die mij steeds gesteund en geholpen heeft.

Samenvatting

In de huidige tijden zijn zowel mensen als organisaties continu betrokken bij een groot aantal activiteiten die gegroepeerd kunnen worden in processen. Vandaag de dag zijn klanten eveneens veeleisender dan ooit waardoor bedrijfsprocessen vaak flexibeler moeten zijn om aan deze specifieke noden van klanten te kunnen voldoen. De meeste real-life bedrijfsprocessen worden dus met andere woorden niet strikt afgedwongen door de ondersteunende informatiesystemen om de nodige flexibiliteit te kunnen behouden. Daarnaast komt ook het digitale tijdperk op dat bedrijven de mogelijkheid geeft om een enorme massa data op te slaan, te verwerken, te analyseren en er vervolgens hopelijk waarde uit te halen. Om sterk te staan in de steeds wijzigende economie is één van de speerpunten van veel bedrijven om de efficiëntie van de voorgenoemde bedrijfsprocessen te verhogen. Deze combinatie zorgt ervoor dat managers en werknemers ondersteund dienen te worden in hun analysewerk. In de laatste decennia hebben onderzoekers met behulp van onder andere *process mining technieken* geprobeerd om die ondersteuning te bieden. Process mining probeert aan de hand van event logs inzichten te verkrijgen in de bedrijfsprocessen en zo mogelijke verbeterpunten bloot te leggen. Deze event logs zijn een groepering van opgeslagen data komende van uitvoeringen van de verschillende activiteiten binnen de processen.

Process mining bevat, volgens de traditionele beschrijving, drie grote disciplines: process discovery, process conformance en process enhancement. Een belangrijke kanttekening hierbij is dat deze opdeling niet meer volledig aanvaard wordt binnen de wetenschappelijke wereld. Het doel van process discovery is om procesmodellen op een geautomatiseerde manier uit event logs, procesdata met andere woorden, te extraheren. De andere twee disciplines gebruiken dit procesmodel vervolgens om bijvoorbeeld real-time monitoring te doen of om verbeteringen te onderzoeken en in te voeren. De kwaliteit van het geëxtraheerde procesmodel is dus uitermate belangrijk om dit op een betrouwbare manier verder te gebruiken voor verschillende doeleinden.

Real-life bedrijfsprocessen spelen zich echter niet af in een ideale omgeving waar steeds een mooi procespad wordt gevolgd. De uitvoering van processen is vaak heel flexibel wat leidt tot vele procesvarianten binnen een 'systeem' dat vaak als één proces wordt gezien. Denk hierbij aan processen in de gezondheidszorg, productontwikkeling of de klantendienst. De process discovery algoritmen, samen met de huidige notaties, kunnen deze complexiteit vaak niet aan en produceren vervolgens onduidelijke en ongestructureerde procesmodellen. Deze ongestructureerde procesmodellen worden ook vaak spaghetti modellen genoemd. Eigenschappen van dergelijke modellen zijn: een groot aantal activiteiten en een groot aantal relaties tussen deze activiteiten.

Process mining is dus een beloftevolle wetenschappelijke discipline die het analyseren van een proces naar een hoger niveau moet brengen. Echter brengen de huidige process discovery algoritmen in combinatie met de gebruikte notaties beperkingen met zich mee. Zo zorgen event logs van flexibele/complexe processen ervoor dat process discovery algoritmen spaghetti modellen creëren. Deze modellen zijn moeilijk begrijpbaar door hun hoge complexiteit en behalen niet de vereiste kwaliteit, meer specifiek de kwaliteitsdimensies precision en simplicity ,om een degelijke analyse van het proces mogelijk te maken. Om process mining naar een volgend niveau te tillen en toepasbaar te maken in de bedrijfswereld, waar vaak complexe en flexibele processen de norm zijn, is er dus een manier nodig om event logs op een betere manier te analyseren die begrijpbare en nuttige procesmodellen als resultaat heeft.

Om dit mogelijk te maken is er de techniek van trace clustering ontstaan. Het doel van trace clustering is tweeledig en bestaat uit het reduceren van de complexiteit van de modellen en het verhogen van de

precisie en de fitness. Bij het toepassen van trace clustering gaat men de event log eerst preprocessen zodat deze vervolgens een betere basis vormt voor de process discovery algoritmen. De trace clustering technieken proberen vaak om de gelijkenissen tussen de verschillende cases/traces te meten en deze vervolgens zo in een aantal clusters te verdelen. Deze clusters stellen in het ideale geval de procesvarianten voor en dienen dan als basis om de event log op te splitsen waarna een process discovery algoritme uit elke cluster een procesmodel kan extraheren. Trace clustering moet ervoor zorgen dat geëxtraheerde procesmodellen uit event data opnieuw verstaanbaar en nuttig zijn voor analisten en andere stakeholders. Door trace clustering toe te passen is een betere analyse mogelijk en stelt men dus eigenlijk de complexiteit van een proces voor in verschillende submodellen.

In deze masterproef wordt er eerst aan de hand van een literatuurstudie een overzicht gegeven van verschillende trace clustering technieken die het mogelijk maken om een event log op een kwalitatieve manier te clusteren alsook trace representation technieken die een grote impact kunnen hebben op de kwaliteit van de uiteindelijke procesmodellen. Daarnaast wordt er ook een vergelijkende analyse gedaan van de verschillende trace clustering technieken. Deze analyse vergeleek in totaal 16 verschillende configuraties van 4 verschillende trace clustering technieken. De trace clustering technieken werden onderling met elkaar vergeleken op basis van verschillende kwaliteitsmaatstaven, met naast de klassieke maatstaven, een specifieke focus voor de complexiteit van de procesmodellen. Verschillende facetten van deze complexiteit werden uitgedrukt aan de hand van 6 verschillende kwaliteitsmaatstaven.

Uit de vergelijkende analyse blijkt dat de trace clustering technieken doorgaans wel een positief effect hebben op de kwaliteitsmaatstaven precision en simplicity. Twee grote lijnen vielen hierbij op. Ten eerste is het effect sterk afhankelijk van de keuze van de trace clustering techniek en van de specifieke configuratie van de parameters. Daarnaast viel ook op dat het effect afhankelijk was van de complexiteit van de gebruikte event log. In de vergelijkende analyse werd bewust gekozen voor drie event logs die verschilde in complexiteit. Eén event log afkomstig van een helpdesk met 14 activiteiten, een andere event log van een aanvraagprocedure binnen een universiteit met 51 activiteiten en een laatste event log afkomstig van een gynaecologische afdeling met maar liefst 624 activiteiten. Het viel op dat de verbetering kleiner werd en zelfs zo goed als verdween naarmate het aantal activiteiten steeg. Dit kan echter ook te maken hebben met de configuratie van de parameters van de verschillende trace clustering technieken.

Deze masterproef had echter ook enkele beperkingen. Wegens praktische redenen, en omwille van het perspectief van een analist, werden in de vergelijkende analyse enkel trace clustering technieken in acht genomen die een publiek toegankelijke plug-in voor ProM (process mining framework) of publiek toegankelijke R/python code ter beschikking stelden. Hierdoor is het aantal trace clustering technieken van de vergelijkende analyse beperkt. Daarnaast was ook de beschikbare computerkracht, die nodig was om real-life event logs volledig te behandelen, te beperkt. In de vergelijkende analyse is er bijgevolg gebruik gemaakt van steekproeven van real-life event logs. Tot slot valt op te merken dat veel trace clustering technieken veel parameters hebben die het uiteindelijke resultaat sterk kunnen beïnvloeden. Sommige technieken vereisen zelfs specifieke detaillistische proceskennis om het de techniek te kunnen gebruiken. Dit kan voordelen hebben omdat procesexperts soms kennis bezitten die essentieel is en niet vervat zit in de data. Echter kan dit ook een zwakte zijn van deze technieken. Een procesexpert heeft namelijk onvermijdelijk een vertekend beeld van het proces. Als dit vertekend beeld meegenomen wordt in de trace clustering technieken riskeert men de clusteralgoritmen in een richting te duwen die mogelijks fout is. Het gevolg hiervan is dat ook de procesmodellen dit beeld impliciet of expliciet zullen bevatten.

Inhoudsopgave

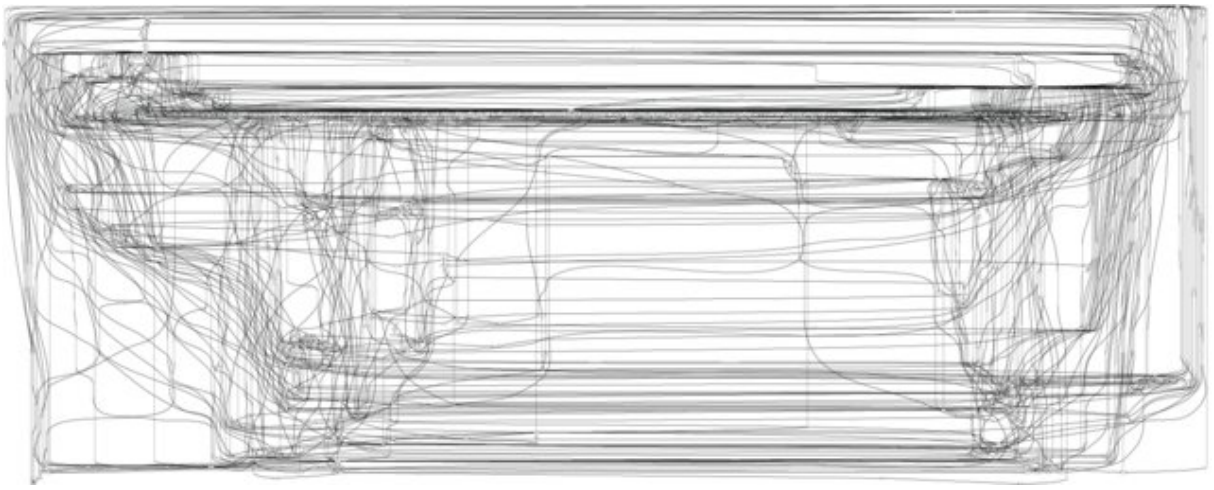
1 Introductie

Vandaag de dag organiseren de meeste bedrijven hun activiteiten in de vorm van bedrijfsprocessen [43, 66]. Dat kan gaan van het produceren van een bepaald product, het leveren van een dienst aan een klant tot het uitkeren van de lonen van de werknemers. Deze processen zijn dus van uitermate groot belang voor elk bedrijf. Langs de ene kant zijn al deze processen samen verantwoordelijk voor het merendeel van de kosten. En langs de andere kant bepaald het verloop van deze processen grotendeels de kwaliteit van de producten of diensten die je als bedrijf levert aan de klanten. Daarbovenop komt nog dat klanten vandaag de dag veeleisender zijn dan ooit en dat de bedrijfsactiviteiten zich doorgaans afspelen in een zeer competitieve, geglobaliseerde markt [43].

In de huidige tijden zijn organisaties continu betrokken bij een groot aantal activiteiten die gegroepeerd kunnen worden in processen. Doordat de klant jaar na jaar meer verwacht van een bedrijf moeten de bedrijfsprocessen vaak flexibeler en efficiënter zijn dan ooit tevoren om aan deze specifieke noden van klanten te kunnen voldoen. Business Process Management blijkt uit onderzoek dan ook één van de meest effectieve manieren om als bedrijf aan deze noden te kunnen voldoen [53, 66, 68]. Eén van de manieren waarop Business Process Management in de praktijk wordt gebracht is het gebruik van informatiesystemen die zich bewust zijn van deze processen. De meeste real-life bedrijfsprocessen worden echter niet strikt afgedwongen door de ondersteunende informatiesystemen om de nodige flexibiliteit toe te laten [62]. Daarnaast komt ook het digitale tijdperk op dat bedrijven de mogelijkheid geeft om een enorme massa data op te slaan, te verwerken, te analyseren en er vervolgens hopelijk waarde uit te halen. Reeds in 2011 werd data gezien als *The next frontier for innovation, competition, and productivity* [44]. Om sterk te staan in de steeds wijzigende economie is één van de speerpunten van veel bedrijven om de efficiëntie van de voorgenoemde bedrijfsprocessen te verhogen. De combinatie van de flexibele processen, de enorme hoeveelheid aan data en de snel veranderende economie zorgt ervoor dat managers en werknemers ondersteund dienen te worden in hun analysewerk [2]. In de laatste decennia hebben onderzoekers met behulp van onder andere *process mining technieken* geprobeerd om die ondersteuning te bieden. Process mining probeert aan de hand van event logs waardevolle inzichten te verkrijgen in de bedrijfsprocessen en zo mogelijke verbeterpunten bloot te leggen. Deze event logs zijn een groepering van opgeslagen data komende van uitvoeringen van de verschillende activiteiten binnen de processen [2].

Eén taak binnen process mining is process discovery. Het doel van process discovery is om procesmodellen op een geautomatiseerde manier uit event logs, procesdata met andere woorden, te extraheren. Real-life bedrijfsprocessen spelen zich echter niet af in een ideale omgeving waar steeds een mooi procespad wordt gevolgd. De uitvoering van processen is, mede door de aard van sommige processen en mede door de noden van de klanten, vaak heel flexibel wat leidt tot vele procesvarianten binnen een 'systeem' dat vaak als één proces wordt gezien. Denk hierbij aan processen in de gezondheidszorg, productontwikkeling of de klantendienst. De process discovery algoritmen, samen met de huidige notaties, kunnen deze complexiteit vaak niet aan en produceren vervolgens onduidelijke en ongestructureerde procesmodellen; zie Figuur 1 waar een ongestructureerd procesmodel van een event log uit de BPI Challenge van 2011 wordt getoond [1]. Dergelijke procesmodellen worden in de literatuur vaak *spaghetti models* genoemd. Eigenschappen van dergelijke modellen zijn: een groot aantal *task nodes* en een groot aantal relaties tussen de nodes [62]. Deze procesmodellen zijn voor analisten quasi onbruikbaar. Men moet dus op zoek naar een andere manier om de waardevolle event data te kunnen analyseren. Eén oorzaak die bijdraagt tot deze ongestructureerde modellen is de flexibiliteit van het proces, of anders benoemd: de diversiteit binnen een dergelijke event log. Met diversiteit binnen een event log bedoelt men typisch dat de verschillende cases binnen een dergelijke event log significant

van elkaar verschillen [62]. Hier zijn cases uitvoeringen van het proces van de start tot het einde.



Figuur 1: Een voorbeeld van een spaghetti model gebaseerd op de event log van de BPI Challenge 2011 [1].

Een vaak gemaakte, en algemeen aanvaarde, assumptie is dat er impliciet procesvarianten verborgen zitten in de event log [46]. Het probleem met deze diversiteit binnen processen is dat deze op het eerste zicht onzichtbaar is. Men kan dus niet makkelijk en eenduidig zeggen welk deel van een event log behoord tot welke procesvariant. Het opsplitsen van een event log in procesvarianten en daarmee het onbegrijpbare terug begrijpbaar te maken is dus niet eenvoudig [46].

Om dit mogelijk te maken is er de techniek van trace clustering ontstaan [62]. Bij het toepassen van trace clustering gaat men de event log eerst preprocessen zodat deze vervolgens een betere basis vormt voor de process discovery algoritmen. De trace clustering technieken proberen vaak om de gelijkenissen tussen de verschillende cases/traces te meten en deze vervolgens zo in een aantal clusters te verdelen. Deze clusters stellen in het ideale geval de procesvarianten voor en dienen dan als basis om de event log op te splitsen waarna een proces discovery algoritme uit elke cluster een procesmodel kan extraheren. Trace clustering moet ervoor zorgen dat geëxtraheerde procesmodellen uit event data opnieuw verstaanbaar en nuttig zijn voor analisten en andere stakeholders [62]. Door trace clustering toe te passen is een betere analyse mogelijk en stelt men dus eigenlijk de complexiteit van een proces voor in verschillende submodellen [46].

Het doel van dit werk is om enerzijds een overzicht te geven van verschillende trace clustering technieken die nodig zijn om een event log op een kwalitatieve manier te clusteren. Anderzijds vergelijkt dit werk verschillende trace clustering technieken, die allen ontworpen zijn in de afgelopen 15 jaar, op een objectieve manier aan de hand van event logs die onderling verschillen in complexiteit. Deze analyse vergelijkt in totaal 16 verschillende configuraties van 4 verschillende trace clustering technieken. De trace clustering technieken werden onderling met elkaar vergeleken op basis van verschillende kwaliteitsmaatstaven met, naast de klassieke maatstaven, een specifieke focus op de complexiteit van de procesmodellen. Verschillende facetten van deze complexiteit werden uitgedrukt aan de hand van 6 verschillende kwaliteitsmaatstaven. Uit de vergelijkende analyse blijkt dat de trace clustering technieken doorgaans wel een positief effect hebben op de kwaliteitsmaatstaven precision en simplicity. Twee grote lijnen vielen hierbij op. Ten eerste is het effect sterk afhankelijk van de keuze van de trace clustering techniek en van de specifieke configuratie van de parameters. Daarnaast viel ook op dat het effect afhankelijk was van de complexiteit van de gebruikte event log.

In de eerstvolgende sectie wordt een beknopt overzicht gegeven over het vakgebied process mining

en hoe men een procesmodel met behulp van enkele maatstaven goed kan beoordelen. Deze kennis is belangrijk om de verschillende trace clustering technieken goed te begrijpen. Als tweede wordt de probleemstelling nader toegelicht in Sectie 3. Daarna worden in Sectie 5 verschillende trace clustering technieken besproken. Naast trace clustering technieken zijn ook de trace representation technieken, die zorgen voor een kwalitatieve input voor de trace clustering, belangrijk. Deze worden daarom ook besproken in Sectie 6. Daarna wordt overgegaan in de vergelijkende analyse van de verschillende trace clustering technieken. Sectie 7 is opgedeeld in een introductie en de methodologie. In deze sectie wordt ondermeer de selectie van de event logs en de trace clustering technieken besproken alsook het verloop van het analyseproces. Na de methodologsiche bespreking van de vergelijkende analyse wordt overgegaan tot de bespreking van de resultaten in Sectie 8. Na de resultaten volgt er nog een conclusie in Sectie 10 waarbij een analyse wordt gedaan van de resultaten van de verschillende trace clustering technieken over de verschillende event logs heen. Tot slot worden ook enkele aanbevelingen voor toekomstig werk en enkele tekortkomingen van dit werk aangehaald in Sectie 9.

2 Process mining

Informatiesystemen die bedrijfsprocessen ondersteunen genereren een grote hoeveelheid data. Deze data wordt ook wel event data genoemd en vormt het startpunt van verscheidene process mining technieken. In deze sectie wordt het gebied van process mining breder toegelicht en wordt er in gegaan op de verschillende disciplines en specifieke terminologie. Daarnaast wordt er ook dieper ingegaan op hoe men in de literatuur de kwaliteit van een procesmodel bekijkt en worden er ook enkele, van de momenteel belangrijkste, process discovery algoritmen toegelicht.

2.1 Algemene introductie

Data science is zowel in de academische wereld als in de bedrijfswereld een hot topic. Een snelle blik op het aantal openstaande vacatures met *data* in de titel of een zoekopdracht in een wetenschappelijke database geeft snel een goed beeld van hoe populair data science eigenlijk is [2]. Bedrijven verzamelen namelijk grote hoeveelheden data van hun klanten, maar ook van hun werknemers, machines en zelfs van hun processen [2]. Onder invloed van onder meer Hammer and Campy [29], die worden gezien als de bedenkers van de shift naar *process-oriented thinking* door hun boek uit 1993, verschuift de focus binnenin bedrijven meer en meer van taken in departementen naar processen voor klanten. Als gevolg veranderen meer en meer systemen, denk aan ERP-systemen of CRM-systemen, in zogenaamde *Process-Aware Information Systems* (PAIS). Deze systemen slaan continu informatie op over de uitvoering van taken en processen binnen een onderneming. Al deze data geeft analisten, mits de juiste technieken voor handen zijn, de mogelijkheid om een beter begrip te krijgen van de processen in de realiteit en zo verder te gaan in de zoektocht naar het meest efficiënte proces. Daarnaast kunnen deze technieken helpen bij het inschatten van risico's in real-time en het onderzoeken van conformance-vragen die bijvoorbeeld optreden bij de vraag of men al dan niet voldoet aan wettelijke vereisten [2].

Process mining reikt deze technieken aan en zorgt ervoor dat analisten de brug kunnen maken tussen traditionele procesanalyse gebaseerd op simulaties en de data-driven analysetechnieken zoals machine learning en data analyse [2]. Process mining vormt met andere woorden de link tussen *traditionele* data science en process science [71]. Hierbij vormen events, geregistreerd in Process-Aware-Information-Systems, de basis van deze tak van data science. Vervolgens worden deze opgeslagen events op verschillende manieren gebruikt om bedrijfsprocessen in kaart te brengen, te analyseren en vervolgens te verbeteren [2, 71]. Process mining wordt in de literatuur traditioneel opgedeeld in drie grote disciplines: process discovery, process conformance en process enhancement. De laatste discipline, process enhancement, is echter niet altijd even goed afgelijnd waardoor deze opdeling meer en meer ter discussie staat [72]. Voor elk van deze disciplines zijn eveneens verschillende perspectieven mogelijk: control-flow perspectief, organisatieperspectief, dataperspectief en tijdperspectief [2].

Process discovery heeft als doel een procesmodel te bouwen gebaseerd op een event log zodat de analist meer inzicht krijgt in het echte verloop van de geanalyseerde processen en in staat is de realiteit te analyseren [5]. Voor deze taak zijn reeds verschillende algoritmen ontwikkeld. Deze algoritmen nemen een event log als input en hebben als output een procesmodel dat de control-flow weergeeft. Maar die vaak verrijkt kan worden met gegevens die betrekking hebben op de andere perspectieven die hier boven reeds zijn aangehaald. Een aantal toonaangevende process discovery algoritmen worden verder besproken in Sectie 2.4. Een goed procesmodel extraheren uit een event log op een automatische manier door middel van een algoritme lijkt echter een moeilijke taak te zijn. Een procesmodel wordt immers pas als goed en waardevol beschouwd als het voldoet aan een aantal vereisten. In [5],

waar de auteurs een vergelijkende studie maken tussen verschillende process discovery algoritmen, en vele andere werken worden de volgende criteria in acht genomen: fitness, precision, generalization en simplicity [13]. De kwaliteit van een procesmodel is uitermate belangrijk voor de verdere analyse van het proces. Deze kwaliteitscriteria worden daarom, met bijhorende specifieke maatstaven, verder besproken in subSectie 2.3.

Process conformance is de volgende grote taak binnen process mining. Het doel van conformance checking is om een procesmodel te vergelijken met event data. De voorgenoemde kwaliteitsmaatstaven komen dus ook hier regelmatig terug. Echter is process conformance breder dan enkel het bekijken van de kwaliteit van een procesmodel. Binnen process conformance gaat men een procesmodel vergelijken met de event data om vervolgens inconsistenties tussen de twee uit te lichten [35]. Process conformance is dus een verzameling van technieken die een procesmodel vergelijkt met een event log met als doel de verschillen en de gelijkenissen tussen enerzijds het procesmodel en anderzijds de event log aan het licht te brengen. Het doel van process conformance is om het gemodelleerde gedrag af te toetsen aan het geobserveerde gedrag in het daadwerkelijke proces en op basis van deze informatie te kijken waar het eventueel beter kan of welke stappen er momenteel minder goed begrepen worden door bijvoorbeeld het management [51]. Nuttige toepassingen van deze discipline zijn bijvoorbeeld het controleren van een bepaald proces in het kader van een certificering of in het kader van een bepaalde wetgeving zoals fraudebestrijding [50].

Process enhancement is een deel van process mining dat volgens een recente literatuurstudie verder opgedeeld kan worden in twee subdisciplines [72] wat ook naar boven komt in de traditionele definitie. De meer traditionele definitie, opgeworpen door Van der Aalst [2], definieert process enhancement als de uitbreiding of verbetering van een bestaand procesmodel waarbij er gebruik wordt gemaakt van informatie afkomstig van de uitvoering van het real-life proces [2, 72]. Technieken die vallen onder deze ruime noemer hebben als gemeenschappelijk doel om het a-priori procesmodel te verbeteren. Dat kan enerzijds door het model te repareren; te herstellen. En anderzijds kan dat ook door het huidige procesmodel uit te breiden op basis van bijkomende informatie. Deze derde categorie binnen process mining, is de categorie die soms ook gezien wordt als de categorie *overige* en waar dus meer technieken onder geduwd worden dan de oorspronkelijke definitie voor ogen had. Een vaak voorkomend voorbeeld van een *overige* techniek is *predictive monitoring* waarbij process mining technieken worden ingezet om de volgende stap van het proces, op basis van real-time en historische data, te gaan voorspellen [72].

2.2 Terminologie

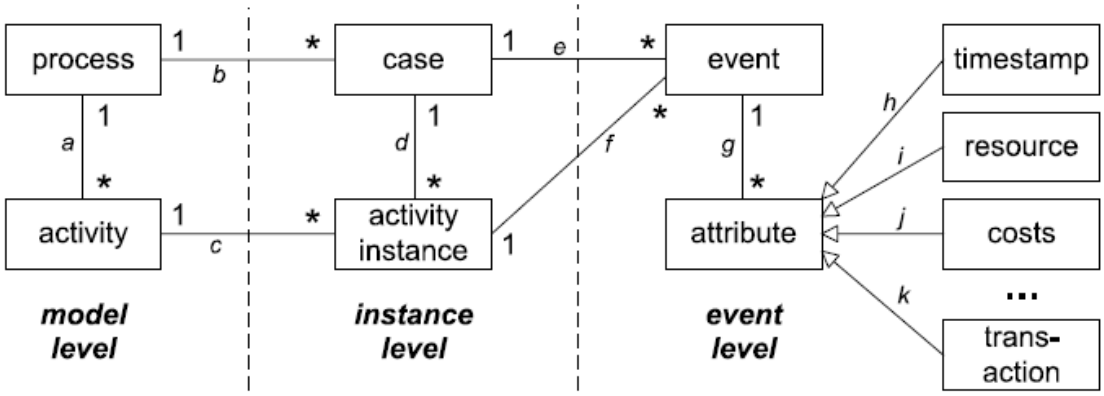
Om de verschillende algoritmen die hierna besproken worden goed te kunnen begrijpen is het nodig dat enkele begrippen, anders genoemd terminologie, wordt toegelicht alvorens deze algoritmen nader besproken worden. In deze sectie wordt de belangrijkste terminologie binnen process mining toegelicht. Hieronder een opsomming van de belangrijkste begrippen die aan bod zullen komen [2]:

- **Een proces** is een verzameling van activiteiten die samen de levenscyclus van een case beschrijven.
- **Een event log** is een verzameling van opgenomen events die dient als input voor technieken binnen process mining. Een event log kan bestaan uit enkel de zogenoemde basisattributen dewelke een case identifier, een activiteit en een attribuut dat het mogelijk maakt om de events te ordenen, doorgaans een timestamp, zijn. Echter bevat een event log vaak ook meer gedetailleerde informatie over verschillende andere aspecten van het proces die handig kunnen zijn bij

een meer gedetailleerde analyse.

- **Een case** is een entiteit die behandeld wordt door het proces dat geanalyseerd wordt. Events refereren steeds naar een case. Voorbeelden van een case zijn: een order van een klant, een verzekeringsclaim, etc.
- **Een activiteit** is een label dat meer duiding geeft wat er gebeurd als de activiteit uitgevoerd wordt. De effectieve uitvoering wordt een activiteitsinstantie genoemd.
- **Een activiteitsinstantie** is een nauw gedefinieerde stap in het proces die uitgevoerd wordt door een *resource*.
- **Events** refereren steeds naar de uitvoering van een activiteit die in meerdere *lifecycles* kan plaatsvinden. Mogelijkheden hiervan zijn: start, voltooiing of annulering.
- **Een trace** is een eindige opeenvolging van events dat een mogelijk pad weergeeft binnen het proces.
- **Een resource** is typisch de persoon of de machine die de activiteit uitvoert.

Om eveneens een breder beeld te schetsen van deze begrippen, worden de relaties tussen de begrippen onderling afgebeeld in Figuur 2 [2]. Aan de hand van het schema wordt duidelijk dat een proces bestaat uit verschillende cases. Een case bestaat op zijn beurt dan weer uit verschillende events die specifiek aan één case gekoppeld zijn en die geordend kunnen worden. Tot slot heeft een event attributen zoals de activiteitsnaam en de timestamp [2].

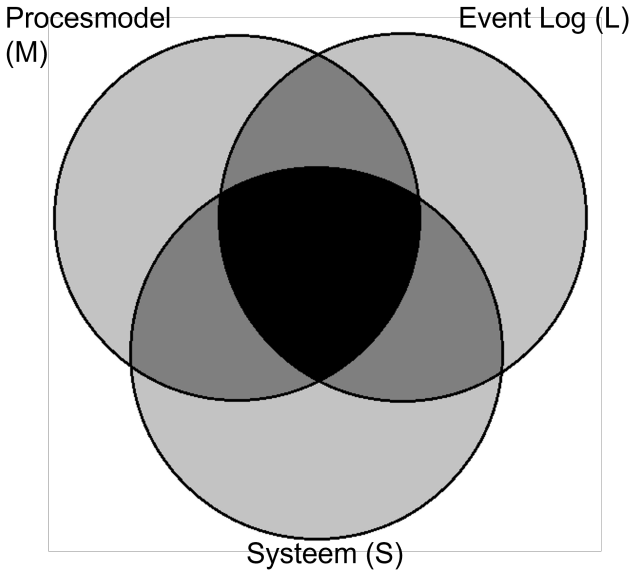


Figuur 2: Onderlinge relaties tussen process mining begrippen [2].

2.3 Kwaliteit van een procesmodel

Over de kwaliteit van een procesmodel, al dan niet afkomstig van een process discovery algoritme, is al veel geschreven. Ondanks dat verscheidene wetenschappers hun hoofd al meermaals hebben gebogen over deze problematiek, is er nog steeds geen eenduidige methode om de kwaliteit op verschillende punten te berekenen. Het evalueren van de kwaliteit van een ontdekt procesmodel is echter wel een belangrijke taak binnen veel analysestappen van process mining. De moeilijkheid bij het bepalen van de kwaliteit van een nieuw procesmodel is dat het trio event log, systeem en procesmodel elk gezien kunnen worden als een bepaalde visie van het procesgedrag die bijna nooit volledig overeenkomen. Het gedrag van deze drie 'werkelijkheden' wordt afgebeeld aan de hand van een venndiagram in Figuur 3 [12]. Zo kan het opgeslagen gedrag dat in de event log vervat zit, meer zijn dan een deelverzameling van het werkelijke gedrag van het proces. Dit ondermeer door data inconsistencies en het

niet accuraat loggen van bepaalde activiteiten. Ook een ontdekt procesmodel laat soms meer gedrag toe dan het gedrag van de event log en bijgevolg ook meer dan het werkelijke systeemgedrag [35]. Men kan, alles bij elkaar opgeteld, dus zeven gebieden van gedrag onderscheiden. Het is belangrijk om te beseffen dat er dus geen manier is om expliciet het gedrag van het systeem te beschrijven. Ten eerste omdat het gedrag van een systeem als oneindig beschouwd kan worden, maar daarnaast ook omwille van het feit dat het altijd mogelijk is dat niet al het mogelijke gedrag vervat zit in de event log [12].



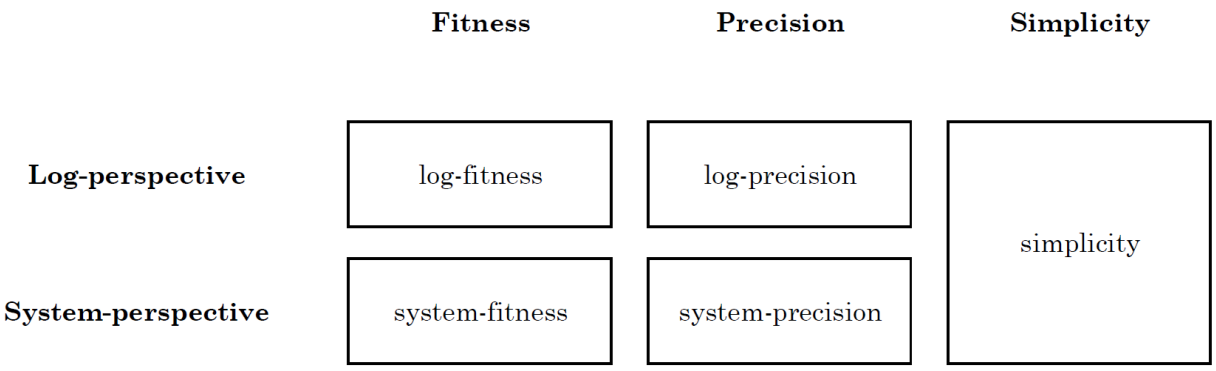
Figuur 3: Een venndiagram dat aantoont dat het gedrag van een procesmodel, een event log en het systeem zowel kan overlappen met als kan verschillen van elkaar [12].

Hoewel men het dus niet altijd eens is over de manier waarop men de kwaliteit van een procesmodel moet benaderen en berekenen praat men typisch wel over vier kwaliteitscriteria. Het doel van process mining wordt immers vaak omschreven als de zoektocht naar een zo goed mogelijk model dat het systeem zo accuraat als mogelijk beschrijft, gebruik makende van het geobserveerde gedrag in de log [12]. De vier kwaliteitscriteria zijn: *precision*, *fitness*, *generalisation* en *simplicity*. In de komende subsecties worden deze verschillende aspecten uitgelegd en nader besproken.

Vooraleer overgegaan wordt tot het bespreken van de verschillende klassieke kwaliteitsmaatstaven dient er een kanttekening gemaakt te worden. In recente literatuur wordt er namelijk gesproken over twee perspectieven wanneer het gaat over de kwaliteit van een procesmodel. Concreet spreekt men dan over het log-perspectief en het systeem-perspectief. Uitgaande van deze gedachtegang moeten de kwaliteitsmaatstaven opgesplitst worden zoals weergegeven in Figuur 4 [35]. Omdat de literatuur nog geen eenduidig antwoord heeft op de vraag of de klassieke opdeling (precision, fitness, generalization en simplicity) al dan niet herdacht moet worden, wordt in deze studie gekozen om verder te gaan met de klassieke opdeling.

2.3.1 Fitness

De fitness dimensie geeft aan hoeveel van het geobserveerde gedrag dat aanwezig is in de event log deel is van het model. Het meet in feite in welke mate de event log het model *fit* [35]. Met het venndiagram uit Figuur 3 in het achterhoofd kan fitness gedefinieerd worden als volgt [12].



Figuur 4: Een weergave van de recentere blik op de kwaliteitsmaatstaven [35].

$$Fitness(L, M) = \frac{|L \cap M|}{|L|}$$

In de vergelijkende analyse van deze paper wordt gewerkt met de *Replay fitness met de alignments method* uit het werk van Bert et al. [7]. Dit is een meer recentere methode dan *token-based fitness* waar in diezelfde paper ook sprake van is. Alignment-based fitness verschilt van token-based fitness in die zin dat de maatstaf de log *aligns* met het model en dat aan de hand van de activiteiten en niet aan de hand van tokens die zich bevinden in het model. Alignment-based fitness gaat aan de hand van een kostfunctie kijken wat het meest waarschijnlijk te volgen pad doorheen het procesmodel is. Deze kostfunctie rekent strafpunten aan voor elke *insertion* en elke *deletion* [35, 7, 12].

2.3.2 Precision

De precision dimensie geeft aan in welke mate het model enkel het gedrag van de event log weergeeft en dus geen gedrag weergeeft dat niet opgenomen is in de event log [35]. Met het venndiagram uit Figuur 3 in het achterhoofd kan precision gedefinieerd worden als volgt [12].

$$Precision(L, M) = \frac{|L \cap M|}{|M|}$$

In de vergelijkende analyse van deze paper wordt gewerkt met de *Align-ETConformance using alignments* precision maatstaf uit [4]. Deze maatstaf vertrekt, net zoals de besproken fitness maatstaf, van alignments tussen de log en het model.

2.3.3 Generalization

Net zoals in andere machine learning technieken is ook hier gevaar voor het fenomeen overfitting. Het is namelijk niet optimaal dat een model perfect overeenkomt met het geobserveerde gedrag en totaal geen ongeobserveerd gedrag toelaat. Als dit wel het geval is kan dat wijzen op een te grote focus op de sample data dat een event log in feite eigenlijk is. De kwaliteitsmaatstaf generalization probeert dit probleem te kwantificeren. In tegenstelling tot precision en fitness is er nog geen algemeen aanvaardde definitie. Generalization kan enerzijds weergeven in welke mate een model overfitting uit de weg gaat [10]. Maar generalization kan ook gedefinieerd worden als de kans dat de volgende, nog niet geobserveerde, case vervat zit in het model [3]. Nog een andere definitie is dat het ontdekte

model het gedrag in de event log zou moeten generaliseren [2]. Een laatste mogelijke definitie die nog aangehaald wordt is dat de kwaliteitsmaatstaf generalization een schatting moet doen in welke mate een model in staat zal zijn om nog niet gezien gedrag te reproduceren [13].

2.3.4 Simplicity

De vierde dimensie is, zoals eerder aangehaald, simplicity. Deze maatstaf gaat, in tegenstelling tot de andere besproken maatstaven, niet het geobserveerde gedrag van de event log vergelijken met het ontdekte model [35]. Simplicity drukt de voorkeur van analisten voor simpelere modellen, in vergelijking met meer complexe modellen, uit. Een complex model heeft namelijk ongewenste effecten op onder andere de correctheid, onderhoudbaarheid en de verstaanbaarheid van het procesmodel[16, 48]. In dit werk krijgt simplicity een bijzondere aandacht aangezien één van de hoofddoelen van trace clustering het reduceren van de complexiteit is [67].

Hoewel de afkeur van analisten voor complexiteit intuïtief klinkt en complexiteit vaak terugkomt in de literatuur, is het moeilijk om één definitie van complexiteit terug te vinden in de literatuur. Het woordenboek van de universiteit van Oxford [17] omschrijft het adjectief complex als een verschijnsel, in dit geval een model, dat bestaat uit veel verschillende en onderling verbonden delen; iets dat niet makkelijk te analyseren of te begrijpen is [14]. Een synoniem is bijvoorbeeld *ingewikkeld* [14]. Een definitie die echter vaker gehanteerd wordt en beter geschikt is voor procesmodelcomplexiteit is de definitie die terugevonden kan worden in de *IEEE Standard Glossary of Software Engineering Terminology* [31]. Hierin definiëren ze complexiteit als de mate waarin het moeilijk is om een proces te analyseren, te begrijpen of uit te leggen [14]. Daarnaast merkt Edmonds ook nog op dat de complexiteit van het echte proces en de complexiteit van het model strikt gescheiden zijn. De complexiteit van een procesmodel kan namelijk voortvloeien uit de gebruikte notatie of uit de beperkingen van de notatie [25].

De verschillende definities maken duidelijk dat de complexiteit van een procesmodel onmogelijk uitgedrukt kan worden aan de hand van één enkele maatstaf. Als men dus een volledig beeld wilt krijgen van de complexiteit van een procesmodel moet men begrijpen dat complexiteit vanuit verschillende standpunten benaderd kan worden [16]. Zo zijn er vier perspectieven die een invloed hebben over de algemene complexiteit van een proces(model). Cardasco heeft het in [15] over complexiteit met betrekking tot de activiteiten, complexiteit gaande over control-flow, data-flow complexiteit en tot slot complexiteit in verband met de resources. Het gevolg hiervan is ook dat simplicity, of het tegenovergestelde complexiteit, niet gevat kan worden in één enkele maatstaf. In dit werk ligt de focus echter op control-flow complexiteit.

In [56] geven de auteurs een overzicht van verschillende maatstaven om de complexiteit van een procesmodel in kaart te brengen. De auteurs bekwamen deze lijst door middel van het uitvoeren van een systematische literatuurreview. In de volgende alinea's worden de volgende maatstaven verder kort besproken: aantal activiteiten en andere elementen, *Halstead-based Process Complexity*, *Coefficient of connectivity*, *Cyclomatic Number*, *Density* en *Control-flow complexity*.

In overeenstemming met een voor de hand liggende maatstaf voor programmaontwikkeling kan men eenvoudigweg kijken naar het aantal activiteiten en control-flow elementen zoals XOR-, OR- en AND-splits [16]. Hoewel dit een heel eenvoudige maatstaf is geven deze verschillende getallen meteen een beeld van hoe groot het beschouwde model is. Concreet in dit werk wordt er gekeken naar het aantal *Nodes* en het aantal *Arcs*. Nodes worden hier gezien als de som van het aantal *places* en het aantal *transitions* in een petri net.

Een andere, meer complexe, maatstaf die wordt voorgesteld in [16] is de *Halstead-based Process Com-*

plexity (HPC). Deze maatstaf gaat de proceslengte, het volume en de moeilijkheid schatten aan de hand van vier waarden. Deze zijn het aantal unieke activiteiten (n_1), het aantal unieke data variabelen (n_2), het totaal aantal voorkomens van activiteiten (N_1) en het totaal aantal voorkomens van data variabelen (N_2). Aan de hand van deze vier waarden kan men vervolgens de proceslengte, procesvolume en de moeilijkheid berekenen:

- Proceslengte: $N = n_1 log_2(n_1) + n_2 log_2(n_2)$
- Procesvolume: $V = (N_1 + N_2) log_2(n_1 + n_2)$
- Moeilijkheid: $D = (n_1/n_2)(N_2/n_2)$

Een andere intuïtieve maatstaf is de *Coefficient of Connectivity* (CoC). Deze maatstaf is de verhouding van het aantal pijlen ten opzichte van het totaal aantal activiteiten, joins en splits [38]. Hoe hoger deze coëfficiënt is, des te complexer het procesmodel

Nog een andere maatstaf die wordt aangehaald in [38] is *Cyclomatic Number*. Deze metriek wordt gedefinieerd als: CN = pijlen - nodes + 1. De *Cyclomatic Number* heeft zijn naam gekregen omdat het het aantal onafhankelijke paden weergeeft die aanwezig zijn in een model [38] en is gebaseerd op McCabe *cyclomatic number* die gebruikt wordt om de complexiteit van software na te gaan [45]. Des te meer mogelijke paden er in een model zitten, des te hoger de *cyclomatic number* en dus des te complexer het model.

Tot slot wordt er graag nog even gefocust op een maatstaf die ligt tussen 0 en 1 en het mogelijk maakt om procesmodellen van verschillende grootteorden te vergelijken met elkaar. Het is de maatstaf *density* en deze wordt gedefinieerd als volgt: aantal pijlen delen door het maximum aantal mogelijk pijlen waarbij deze laatste (de noemer) berekend wordt door het aantal nodes te vermenigvuldigen met het aantal nodes min één. Density is dus: pijlen/(nodes*(nodes-1)) [47].

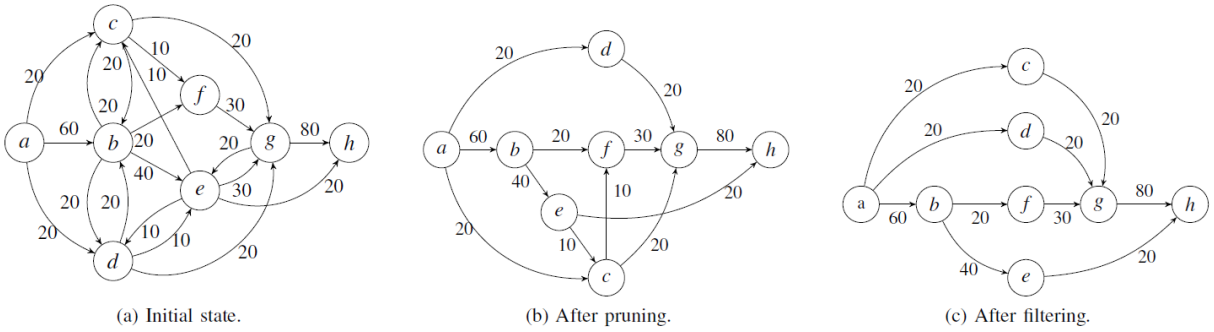
Veel van bovenstaande simplicity maatstaven worden berekend aan de hand van het aantal *arcs* en *nodes*. Om het blikveld te verruimen wordt in deze studie ook nog een andere maatstaf gebruikt die eveneens beschikbaar is in PM4PY [55]. De auteurs van PM4PY hebben gekozen om de *inverse arc degree* te integreren uit een paper van Blum [8] en die eerder beschreven werd in [61]. De *inverse arc degree*, die verder in de paper *Simplicity* wordt genoemd naar analogie met de benaming in PM4PY, wordt in verschillende stappen berekend. Eerst wordt het gemiddeld aantal pijlen verbonden aan een node berekend. De simplicity maatstaf kan dan gedefinieerd worden als volgt.

$$Simplicity(M) = \frac{1}{1 + max(gemiddeld\,aantal\,pijlen - k, 0)}$$

2.4 Process discovery algoritmen

Process discovery is de verzameling van technieken waarbij er, gegeven een event log bestaande uit geobserveerd gedrag, op zoek wordt gegaan naar een procesmodel dat het best dit geobserveerde gedrag beschrijft. Echter, gezien de moeilijkheid van dit probleem, is er momenteel nog geen algoritme voor handen dat een *sound* en *fitting model* garandeert met een voldoende hoge precisie en fitness. Een *sound* model is een model dat vrij is van *deadlocks*, paden die ervoor zorgen dat het procesmodel niet eindig is, waarentegen een *fitting* model betekent dat al het geobserveerde gedrag van de event log vervat zit in het geconstrueerde procesmodel [5]. Om een volledig beeld te schetsen wordt in deze sectie een overzicht gegeven van enkele vooraanstaande process discovery algoritmen. Meer bepaald: split miner, fodina en de inductive miner [5].

Split miner Een algoritme dat sinds de publicatie veel aandacht heeft gekregen, is het algoritme *Split Miner* [6]. Split miner wordt beschreven als een process discovery algoritme dat eenvoudige procesmodellen met een beperkte vertakkingscomplexiteit en consistent een hoge fitness, precisie en generalisatie verwezenlijkt waarbij er een balans wordt gezocht tussen deze kwaliteitscriteria. Het split miner algoritme produceert een BPMN-model in 5 opeenvolgende stappen. De eerste stap is het opbouwen van een DFG; een direct following graph. Een voorbeeld van een DFG kan teruggevonden worden in Figuur 5a [6]. Een DFG geeft de *direct follows* relaties tussen de verschillende aanwezige elementen weer. Bij de tweede stap van dit algoritme wordt de DFG geanalyseerd om reeds in het begin *self-loops* en *short-loops* te detecteren. Daarnaast gaat het algoritme in deze tweede stap ook nog op zoek naar *concurrency* relaties. In DFG termen betekent dat er een pijl is van activiteit a naar b en vice versa. Dit fenomeen betekent dat causaliteit en concurrency met elkaar verward worden. Causaliteit betekent in deze context dat activiteit *a* gevolgd wordt door aciviteit *b* en niet andersom. Terwijl concurrency betekent dat activiteiten *a* en *b* in beide richtingen kunnen voorkomen. Als dit fenomeen wordt waargenomen worden de pijlen tussen beide activiteiten verwijderd van de DFG wat meteen zorgt voor de uitkomst van stap twee: een *pruned DFG* (PDFG) die wordt afgebeeld in Figuur 5b. Vervolgens wordt de PDFG in de derde stap gefilterd om de complexiteit en de accuraatheid onder controle te houden. Deze filtering stap bestaat uit drie delen. Eerst gaat men kijken of elke node op een pad ligt van een start node naar een finale node. Vervolgens wordt het aantal pijlen tussen activiteiten geminimaliseerd. Tot slot kijkt het algoritme naar de som van de frequenties op elk pad om de fitness te maximaliseren. Deze drie substappen kunnen jammer genoeg niet allemaal gelijktijdig geoptimaliseerd worden waardoor het algoritme een trade-off zoekt tussen de verschillende filterstappen. Het proces van het opstellen van zo een *filterd pruned DFG* wordt weergegeven in Figuur 5 [6]. Na de filterstap wordt in stap vier en stap vijf respectievelijk split en join gateways toegevoegd worden aan het procesmodel [6].



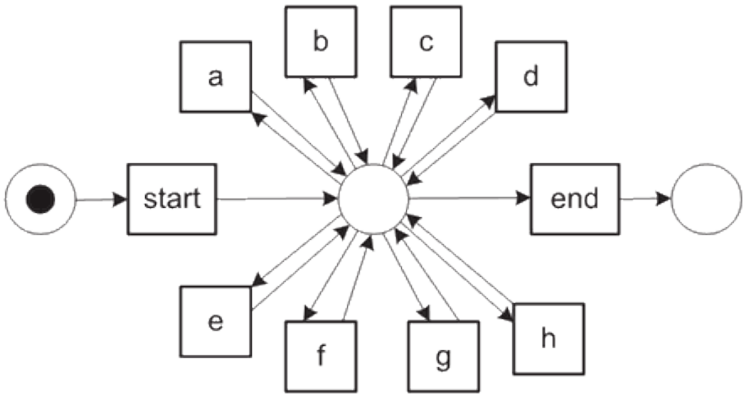
Figuur 5: De evolutie van een *direct-follows graph* doorheen het split miner algoritme [6].

Fodina Verschillende process discovery algoritmen onderscheiden zich doorgaans door te focussen op een bepaald aspect van het uiteindelijke procesmodel. Zo focuste het split miner algoritme voornamelijk op het zoeken naar een goed evenwicht tussen enerzijds de complexiteit van het uiteindelijke procesmodel en anderzijds de accuraatheid van dat procesmodel. Fodina [9], dat in feite een verbetering is van de klassieke heuristic miner, voegt een aantal elementen aan het algoritme toe dat er voor moet zorgen dat het algoritme robuuster is voor *noisy data*, dat het algoritme duplicate activiteiten kan terugvinden en dat het de mogelijkheid geeft aan de eindgebruiker om enkele configuratieopties in te stellen [9]. Noisy data wordt, in de context van dit algoritme, geïnterpreteerd als louter het weinig voorkomen van een activiteit op een bepaalde plaats in het proces. Men werkt dus met een *duplicate task threshold*. Deze threshold zal ervoor zorgen dat bepaalde duplicate activiteiten worden verwijderd uit het procesmodel en samen worden gevoegd met de duplicate activiteit met de hoogste

frequentie. Het algoritme bestaat in totaal uit 8 stappen, waarvan er drie optioneel zijn. De eerste stap transformeert de event log naar een task log waarbij contextuele informatie wordt gebruikt om duplicaten te ontdekken. Vervolgens, in stap twee, gaat het algoritme het aantal basisrelaties, zoals direct following relations, tussen de activiteiten tellen. Van stap drie tot stap zes wordt de DFG opgesteld waarbij er gebruik wordt gemaakt van *dependency measures* zoals het aantal basisrelaties tussen bepaalde activiteiten. Vervolgens wordt er in stap zeven optioneel op zoek gegaan naar *long-distance dependencies*. Tot slot wordt er in stap acht op zoek gegaan naar de semantische informatie. Hiermee worden onder andere de split en join gateways bedoeld [9].

Inductive miner Als men in de literatuur verwijst naar de *Inductive Miner* [40], dan verwijst men eigenlijk naar een verzameling van process discovery technieken die *process trees* bouwen. Deze process trees kunnen echter wel makkelijk omgevormd worden naar andere notaties zoals petri nets en BPMN-modellen. Het framework dat rond deze process discovery technieken gebouwd is kan makkelijk uitgebreid worden waardoor er in deze familie verschillende technieken bestaan die elk een eigen focus hebben. Sommige varianten focussen op het behandelen van infrequent gedrag waarentegen andere modellen focussen op precisie [2]. Het basisalgoritme gebruikt, net zoals vele andere process discovery technieken, een *directly-follows graph* dat de *direct follows* relatie weergeeft. Het inductive miner algoritme splitst de initiële event log op in kleinere sublogs op basis van drie criteria: de *direct follows* relaties, de verzameling van startactiviteiten en de verzameling van eindactiviteiten. Om te bepalen hoe de initiële event log opgesplitst moet worden, gaat het algoritme zoeken naar vier soorten *cuts*. In het algoritme worden volgende *cuts* beschouwd: *exclusive-choice cuts*, *sequence cuts*, *parallel cuts* en *redo-loop cuts*. Deze *cuts* komen overeen met de vier process tree operatoren: XOR, OR, AND en direct-follow. Deze vier *cuts* hebben een bepaalde prioriteit. Om een event log op te splitsen gaat het algoritme eerst kijken of er niet triviale *exclusive-choice cuts* bestaan. Als deze bestaan, wordt deze maximaal toegepast waardoor de event log opgesplitst wordt in verschillende sublogs. Bestaan deze niet dan gaat men vervolgens kijken naar de aanwezigheid van niet triviale *sequence cuts* en naargelang de event log opsplitsen in sublogs. Zijn ook deze niet aanwezig dan gaat men achtereenvolgens kijken naar niet triviale *parallel cuts* en *redo-loop cuts*.

Het opsplitsten van een event log of het verder opsplitsen van sublogs gaat door totdat een zogenoemde *base case*, een sublog met slechts één activiteit, bereikt wordt. Als er geen niet triviale *cuts* gevonden worden, creëert men een *fall-through*. Het deel van de event log of de sublog dat niet opgesplitst kan worden, wordt dan voorgesteld door middel van een *flower model*. Een *flower model* is een procesmodel dat elk gedrag toelaat. Een voorbeeld kan gevonden worden in Figuur 6 Deze optie dient als laatste mogelijke oplossing om de fitness van het model te optimaliseren maar kan wel zorgen voor een verlaagde precisie. Het is belangrijk om op te merken dat de inductive miner steeds een *sound* procesmodel garandeert waardoor dus steeds de hele event log afgespeeld kan worden op het model [40]. Het basis algoritme dat hier net kort werd toegelicht en geïntroduceerd werd door Leemans et al. [40] kan, zoals eerder aangehaald, makkelijk uitgebreid worden. De afgelopen jaren is het algoritme dan ook uitgebreid om bijvoorbeeld infrequent gedrag beter te behandelen [41] of incompleet gedrag te incorporeren [42].



Figuur 6: Een voorbeeld van een flower model. Een procesmodel dat elk gedrag toelaat [2]

Naast bovenstaande process discovery algoritmen bestaan er ook nog andere process discovery algoritmen zoals de α-miner, de Genitc miner, de Evolutionary Tree Miner, de Heuristic Miner en anderen [5]. Een recente review en benchmark studie van verschillende process discovery algoritmen werd uitgevoerd door Augusto et al. [5].

3 Probleemstelling

Process mining is, zoals bleek uit de voorgaande sectie, een beloftevolle wetenschappelijke discipline die het analyseren van een proces naar een hoger niveau moet brengen. Echter brengen de huidige process discovery algoritmen in combinatie met de gebruikte notaties beperkingen met zich mee. Zo zorgen event logs van flexibele processen ervoor dat process discovery algoritmen spaghetti modellen creëren. Deze modellen zijn moeilijk begrijpbaar door hun hoge complexiteit en missen de vereiste precisie en fitness om een degelijke analyse van het proces mogelijk te maken. Om process mining naar een volgend niveau te tillen en toepasbaar te maken in de bedrijfswereld, waar vaak complexe en flexibele processen de norm zijn, is er dus een manier nodig om event logs op een betere manier te analyseren die begrijpbare en nuttige procesmodellen als resultaat heeft.

Een mogelijke oplossing voor het probleem van spaghetti modellen, die de laatste tijd meer en meer geopperd wordt binnen de literatuur, is trace clustering. Het doel van trace clustering is tweeledig en bestaat enerzijds uit het reduceren van de complexiteit van de modellen en anderzijds uit het verhogen van de precisie en de fitness. Echter zijn er momenteel tal van dergelijke technieken beschikbaar die enerzijds overwelmend kunnen zijn voor de analist in aantal en anderzijds het werk alsnog bemoeilijkt door het feit dat moeilijk te achterhalen is welke trace clustering techniek de beste is voor de situatie van de desbetreffende analist.

Dit werk schetst daarom eerst een overzicht van bestaande trace clustering technieken en gaat dieper in op belangrijke karakteristieken inclusief trace representation technieken die een belangrijke rol blijken te spelen bij de clusterresultaten. Dit om een duidelijk overzicht te geven van de reeds bestaande technieken. Daarnaast wordt er in dit werk ook een vergelijkende analyse tussen de trace clustering technieken uitgevoerd op basis van verschillende real-life event logs die het mogelijk moet maken om de verschillende technieken op een objectieve manier met elkaar te gaan vergelijken.

Dit artikel geeft de analist dus meer duidelijkheid over de mogelijkheden van trace clustering en hoe men kan omgaan met meer complexe processen die doorgaans afkomstig zijn van een flexibele werkomgeving.

4 Zoekstrategie

Om meer duidelijkheid te verkrijgen omtrent het onderwerp trace clustering technieken en de plaats hiervan binnen het bredere process mining, is het belangrijk dat de zoektocht naar informatie verloopt volgens een degelijke zoekstrategie. In deze sectie wordt de gehanteerde zoekstrategie toegelicht.

Building block search method Om de relevante literatuur over trace clustering technieken te selecteren is het belangrijk om een goede zoekstrategie te definiëren en aan te houden. Het grootste deel van de papers werd gevonden aan de hand van een aantal kernwoorden en de synoniemen daarvan. De kernwoorden werden zorgvuldig gedefinieerd, aan de hand van onder meer sleutelwoorden van gekende papers over het onderwerp, waarna er gebruik gemaakt werd van de *building block search method* om de effectieve zoekopdrachten uit te voeren [37]. Deze methode helpt om verschillende zoektermen te formuleren. Hierbij worden de verschillende termen opgedeeld in facetten; zogenaamde *blocks*. Deze facetten zijn groeperingen van termen met dezelfde betekenis. Vervolgens worden er combinaties gemaakt tussen verschillende begrippen met behulp van de gekende operatoren AND, OR en NOT. De gebruikte zoektermen, opgedeeld per facet, worden weergegeven in Tabel 1.

Tabel 1: Zoektermen per facet volgens de *building block search method*

Facetten	**Zoektermen**
Process Mining	Process Mining, process discovery, process, Business Process Management
Trace clustering	trace clustering, process instance clustering, sequence clustering, clustering

Zoekopdrachten Met de gedefinieerde zoektermen kon er vervolgens aan de slag worden gegaan om bepaalde zoekopdrachten uit te voeren. Hierbij werden de zoektermen van facetten op zichzelf gebruikt voor een zoekopdracht, maar werden er ook verschillende facetten gecombineerd. Zo werd er gezocht naar technieken voor trace clustering binnen het onderzoeksdomein process mining. Om goede resultaten te bekomen werden de verschillende zoektermen van de facetten *process mining* en *trace clustering* in zoekopdrachten gecombineerd met de operator AND.

Inclusie- en exclusiecriteria Naast het goed definiëren van zoektermen, is het ook belangrijk om een aantal inclusie- en exclusiecriteria te definiëren. Deze criteria kunnen gezien worden als een extra garantie voor de kwaliteit van de gevonden en later geselecteerde literatuur. De verschillende criteria zijn:

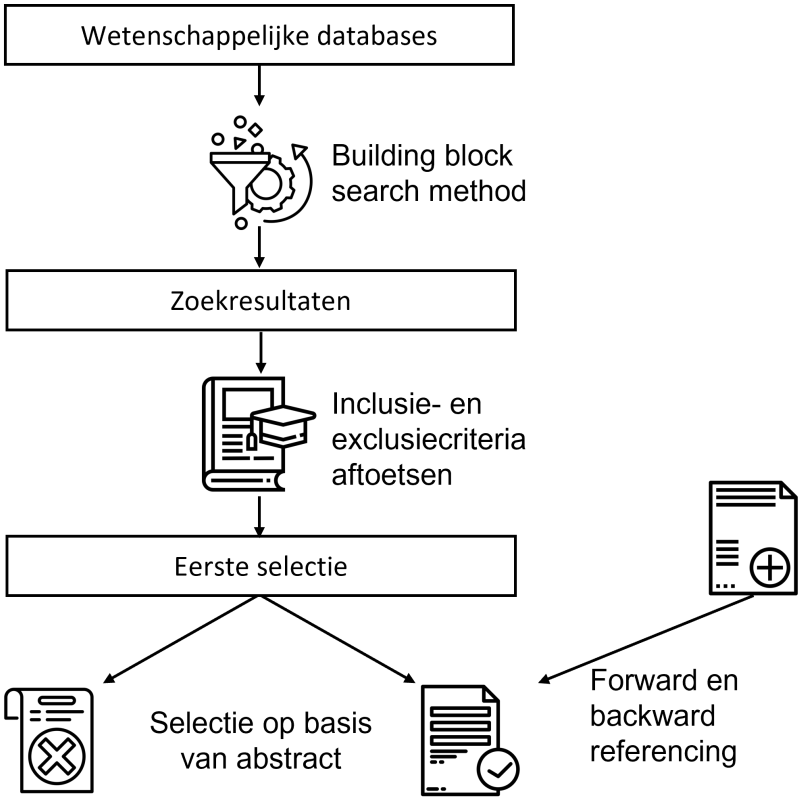
- Het werk, later wetenschappelijk werk of paper genoemd, moet gepubliceerd zijn in een wetenschappelijk tijdschrift of moet opgenomen zijn in de proceedings van een wetenschappelijke conferentie of het gaat om een wetenschappelijk boek
- De paper moet Engelstalig zijn
- Als het gaat om een paper, moet de paper een peer-review doorstaan hebben
- Het wetenschappelijk werk bespreekt een specifieke methode omtrent trace clustering of bespreekt trace clustering in een breder perspectief
- Informatie van het 'wilde' internet komt niet in aanmerking
- De volledige tekst (of het hoofdstuk in kwestie) moet vrij, of dankzij de Universiteit Hasselt, beschikbaar zijn

Backward reference citing Naast deze zoekstrategie werd er ook gebruik gemaakt van *backward en forward reference searching* [54] om een volledig beeld te krijgen van de relevante literatuur. Backward reference searching houdt in dat men de geciteerde bronnen in een gevonden paper gaat onderzoeken. Bij forward reference searching daarentegen worden papers toegevoegd aan de referentielijst die refereren naar een reeds gevonden paper.

Databanken Wetenschappelijke literatuur kan in verschillende databanken gevonden worden. Daarom is het noodzakelijk om te zoeken in verschillende databanken. Voor dit onderzoek is gebruikt gemaakt van bekende databanken die als betrouwbaar worden geacht. Hieronder een opsomming van de gebruikte databanken: Google Scholar, ResearchGate, IEEE, UHasselt Discovery (een verzameling van verschillende wetenschappelijke databanken) en ScienceDirect

Selectie De geïdentificeerde artikelen werden al dan niet geselecteerd op basis van de inhoud van het abstract. Een paper werd opgenomen in de selectie als de paper duidelijk betrekking had op het clusteren van instanties (traces) van bedrijfsprocessen of het algemeen meer inzicht gaf in de wereld van process mining in een flexibele omgeving.

De bovenstaande methodologie bespreekt de zoektocht naar papers met als onderwerp trace clustering. De volgens deze methodologie gevonden literatuur vormt dan ook de basis voor de vergelijkende analyse van verschillende trace clustering technieken die beschreven worden in Sectie 7. Naast deze geselecteerde artikelen is het ook belangrijk om andere wetenschappelijke literatuur mee in rekening te nemen om een volledig beeld te krijgen van enerzijds process mining en anderzijds de problematiek van zogenoemde spaghetti models. Daarom zal de referentielijst uitgebreider zijn dan de selectie op basis van bovenstaande zoekmethode. Bovenstaande zoekmethode heeft echter wel geleid tot de basis van deze paper. De zoekmethode wordt visueel weergegeven in Figuur 7.



Figuur 7: De zoekstrategie visueel voorgesteld.

5 Trace clustering technieken

Om de procesmodellen die voortvloeien uit een process discovery algoritme actief en nuttig te kunnen gebruiken in een reële setting is het, zoals in de vorige sectie meermaals beschreven, belangrijk dat het geëxtraheerde procesmodel goed scoort op de algemeen aanvaarde kwaliteitscriteria. De reeds ontwikkelde process discovery algoritmen hebben echter problemen met de meer flexibele processen uit de echte wereld. Concreet produceren de voorgenoemde algoritmen vaak spaghetti-achtige modellen, alsook modellen die zeer complex zijn en een lage precisie hebben [71]. Er is dus nood aan nieuwe technieken die voor betere procesmodellen zorgen. Trace clustering technieken kunnen hier een mogelijke oplossing voor zijn. Doorheen de jaren zijn verschillende trace clustering technieken ontstaan die het mogelijk maken om de traces van een bepaalde event log te clusteren en zo voor te bereiden op de volgende stap: process discovery. Bij het toepassen van trace clustering gaat men de event log eerst preprocessen zodat deze vervolgens een betere basis vormt voor de discovery algoritmen [73]. Hierdoor is een betere analyse mogelijk. Men stelt dus eigenlijk de complexiteit van een proces voor in verschillende submodellen [71]. Voordelen hiervan zijn legitiem: begrijpbare procesmodellen, procesvarianten worden duidelijk gesplitst en de uiteindelijke procesmodellen zijn minder complex.

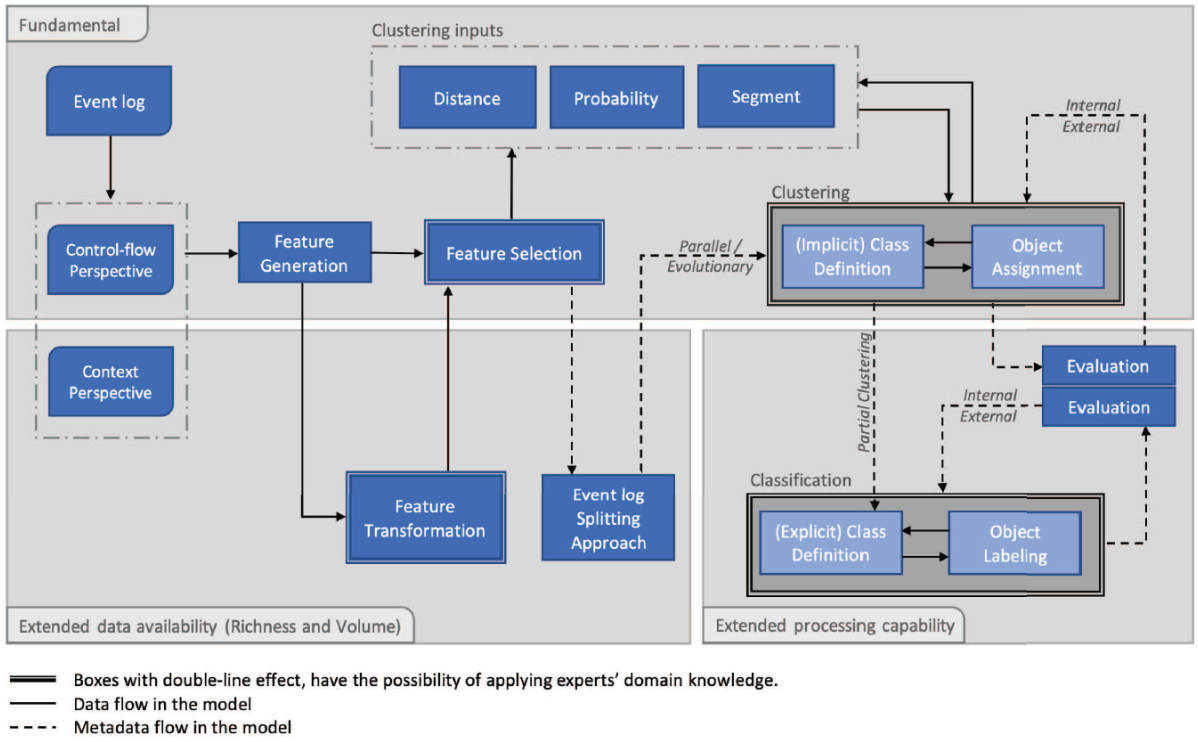
In de literatuur zijn, op basis van de doelstelling van de techniek, twee grote groepen te onderscheiden. Enerzijds zijn er clustering technieken die proberen om een event log op te splitsen in duidelijk te onderscheiden procesvarianten. Anderzijds zijn er clustering technieken die het reduceren van de modelcomplexiteit als hoofddoel hebben en dus proberen om de begrijpbaarheid en de leesbaarheid te verhogen [62]. In deze sectie en de verschillende subsecties worden verschillende trace clustering technieken besproken. Eerst wordt er echter een algemeen framework aangehaald om de verschillende stappen binnen een dergelijk trace clusteringproject goed te kunnen kaderen.

5.1 Een algemeen framework voor trace clusteringprojecten

Ongeacht het doel of de gebruikte methode, volgen de verschillende trace clustering technieken een min of meer vast pad van verschillende stappen. Doorheen dat pad kunnen volgens Zandkarimi et al. [73], op basis van een literatuurstudie van 103 papers, drie grote groepen van stappen onderscheiden worden: fundamentele stappen die altijd terugkomen, stappen die enkel voorkomen als de event log extra data ter beschikking heeft en tot slot stappen die een bepaalde rekenkracht nodig hebben en dus enkel voorkomen wanneer er voldoende rekenkracht ter beschikking is. Deze verschillende groepen en hun stappen worden overzichtelijk weergegeven in Figuur 8 [73].

In de eerste grote blok, de blok *fundamental*, wordt duidelijk dat elk trace clusteringproject start met een event log. Hierbij wordt meteen een opsplitsing gemaakt tussen het control-flow perspectief en het perspectief waar context, met andere woorden extra informatie over onder andere personeel en attributen die gelinkt zijn aan een specifieke case, mee in rekening wordt genomen. Alle stappen in de grijze zone van de blok *fundamental* kunnen uitgevoerd worden met een zogenaamde basis event log zonder extra attributen. Als de event log in kwestie wel beschikt over extra attributen is het aan te raden deze ook te gebruiken en het te volgen pad uit te breiden naar de stappen weergegeven in het blok *extended data availability* [73].

Met behulp van de stappen in de zone *extended data availability* kan de analist beter gebruik gaan maken van de aanwezige data. Met behulp van *feature transformation* gaat de analist impliciete informatie omvormen tot expliciete informatie die vervolgens gebruikt kan worden als extra contextinformatie. Deze extra informatie kan ervoor zorgen dat de clusterresultaten verbeterd worden [73].



Figuur 8: Algemeen framework voor trace clustering technieken in process mining [73].

Een volgende stap binnen de blok *fundamental* is het bepalen van de inputs voor de latere clustering algoritmen. Trace clustering technieken proberen gelijkaardige traces te groeperen zodat er in een volgende stap eenduidige procesmodellen uit geëxtraheerd kunnen worden. Idealiter weerspiegelen deze verschillende procesmodellen dan elk een procesvariant. Om de gelijkheid van verschillende traces te beoordelen kan men gebruik maken van verschillende soorten inputs. Een eerste mogelijke, en veruit de populairste optie, is het gebruik van afstandsmaatstaven [73]. Denk hierbij aan de euclidische afstandsmaatstaf of de Jaccard afstandsmaatstaf. Naast afstandsmaatstaven kan men ook nog werken met probability theories. Hierbij bouwt men Markov-modellen op waarbij het model de kans berekent dat een trace waargenomen wordt, en wordt op deze manier toegekend aan een cluster. Een derde mogelijke aanpak voor de clusteralgoritmen is het gebruik van segmentaties. Bij deze methode maakt men gebruik van een rooster of een net waarbij men ofwel begint met individuele traces en vervolgens probeert zoveel mogelijk naburige traces te groeperen. Ofwel begint men met de gehele event log en probeert men om de event log te splitsen tot het vooraf gedefinieerde doel bereikt is. De plaats waar het algoritme de event log gaat splitsen (of samenvoegen) kan bepaald worden door middel van verschillende criteria. Eén voorbeeld, dat verder uitgewerkt is in [52] door Nguyen et al., is door het toepassen van het *minimum cut*-principe berekend door het Ford–Fulkerson algoritme. Een andere optie is om het *Nearest Neighbor Clustering* algoritme toe te passen op het rooster zoals uitgewerkt is door Kanj et al. [36].

Nadat de inputs bepaald zijn, kan de keuze van het clustering algoritme gemaakt worden. Drie benaderingen kunnen voor deze keuze onderscheiden worden: *full clustering*, *full classification* en *partial clustering* [73]. Bij de eerste benadering, full clustering, zal het algoritme eerst de initiële clusters detecteren en vervolgens alle traces toewijzen aan deze clusters. Echter laat deze groep van algoritmen ook toe dat kennis van een expert het algoritme helpt, meer bepaald het bepalen van de initiële clusters. Een expert kan bij deze benadering met andere woorden zijn kennis van het proces gebruiken om initiële clusters aan te wijzen die het algoritme vervolgens zal gebruiken als richtlijn. Bij de volgende

benadering, full classification, zal een expert steeds zijn kennis gebruiken om de initiële clusters te bepalen. Dit zal gebeuren door het manueel groeperen van traces. Deze initiële clusters worden vervolgens opnieuw gebruikt als leidraad bij het clusteren van de overige traces. In de partial clustering benadering, wordt een combinatie van clustering en classification technieken gebruikt. Concreet zullen de initiële clusters bepaald worden door een combinatie van de kennis van een expert en het resultaat van een eerste clustering die uitgevoerd is op een beperkt deel van een event log. Vervolgens zal een classificatie algoritme de overige traces die nog aanwezig zijn in de event log verdelen over de initiële clusters [73].

Tot slot wordt er in het framework van Zandkarimi et al. [73] ook nog aandacht geschonken aan een andere optionele blok, namelijk de blok *extended processing capability*. In de paper worden in deze blok stappen behandeld die extra evaluatiecriteria toevoegen aan de gebruikte clustering en classificatie-algoritmen. Normaliter hebben deze algoritmen reeds hun eigen ingebouwde evaluatiecriteria om te bepalen hoe goed of slecht het clusterresultaat is. Echter kunnen, naar gelang de toepassing, andere evaluatiecriteria eveneens nuttig zijn. Als men deze algoritmen gaat toepassen in een process mining context en meer bepaald bij trace clustering, is het namelijk niet vreemd om meer te focussen op het doel van deze trace clustering technieken waar de clustering algoritmen uiteindelijk zijn in ingebed. Specifiek kan men dan gaan kijken in welke mate het algoritme de kwaliteit van de procesmodellen verbetert met een voorname focus op verminderde complexiteit, een hogere fitness en een hogere precision. Door deze evaluatiecriteria toe te voegen, helpt men de clusteralgoritmen in hun zoektocht naar de meest ideale clusters. Deze verschillende doelstellingen combineren in één werkend algoritme kan aan de hand van *multi-objective decision making* benaderingen. Voorbeelden hiervan zijn *weighted sum*, *disaggregation method* of *Pareto optimality* [73].

Dit framework wil analisten een duidelijk overzicht geven van de verschillende mogelijke stappen in een trace clustering project waarbij rekening wordt gehouden met verschillende aspecten zoals de beschikbaarheid van extra attributen of de beschikbaarheid van al dan niet uitgebreide computerkracht. In deze paper heeft dit framework zijn plaats gekregen om een overkoepelend beeld te geven van hoe de trace clustering technieken te werk gaan, welke inputs er nodig zijn en wat de mogelijke uitbreidingen zijn. In de volgende subsectie wordt dieper ingegaan op enkele concrete trace clustering technieken.

5.2 Bespreking van de trace clustering technieken

In deze subsectie worden verschillende trace clustering technieken nader besproken en de desbetreffende voor- en nadelen toegelicht. De bespreking van de clustering technieken gebeurt op een chronologische manier zodat een mogelijke evolutie in dit wetenschappelijk gebied duidelijk wordt.

5.2.1 Profiles - 2008

Eén van de eerste en meest impactvolle papers met betrekking tot trace clustering is de paper van Song et al. [62] uit 2008. De auteurs stellen dat de toenmalige process discovery algoritmen niet geschikt zijn voor het behandelen van event logs afkomstig van flexibele, complexe processen. De techniek die Song et al. voorstellen is een *divide-and-conquer* methode gebaseerd op een verzameling van *profiles* die elk een aantal features voor elke case mee in rekening nemen. Deze *profiles* vormen, over de gehele event log bekeken, feature matrices waarop vervolgens afstandsmaatstaven worden toegepast die de relatieve afstand tussen twee cases berekenen [62]. Tot slot worden data clustering algoritmen gebruikt om gerelateerde cases te groeperen in clusters.

Het meest onderscheidende van deze paper is het gebruik van *profiles*. Deze *profiles* proberen op een toepasselijke manier de gelijkheid van traces te bepalen aan de hand van de gegeven informatie in de event log. In een event log zit vaak veel gestructureerde informatie, maar zit ook impliciete informatie verscholen zoals het aantal events in een bepaalde trace. In de benadering van Song et al. [62] worden traces dus afgebeeld aan de hand van *profiles* waarbij een *profile* niet meer is dan een verzameling van gerelateerde items die de trace beschrijven vanuit een specifiek standpunt. Hierbij is elke item een maatstaf. Laat het duidelijk zijn dat elke trace meerdere *profiles* kan hebben. Doorgaans gebruikte *profiles* zijn bevoorbeeld de *activity profile* waarbij de verschillende activiteiten van een trace geteld worden en vervolgens weergegeven worden in een vector waarbij de volgorde van de elementen vooraf gedefinieerd wordt. Een andere vaak gebruikte *profile* is de *transition profile* waar de focus meer ligt op de opeenvolging van de activiteiten en niet zozeer op de aanwezigheid van een activiteit. Tal van andere nuttige *profiles* kunnen, afhankelijk van de aanwezige data, gevonden worden in event logs [62]. Deze profiles dienen vervolgens als input voor de berekening van de 'afstand' tussen twee verschillende traces. Tot slot kan de afstandsmatrix gebruikt worden als input voor een clusteralgoritme. Song et al. [62] bespreken in hun paper volgende methoden:

- **K-means**: de meest gebruikte clusteringtechniek; maakt *k* clusters door de data te verdelen in *k* groepen
- **Quality Threshold Clustering**: oorspronkelijk ontwikkeld voor bio-informatica; hoewel computationeel zwaarder is deze techniek wel voorspelbaar. Deze clustermethode wordt geleid door een *quality threshold* die de maximum diameter van een cluster bepaald.
- **Agglomerative Hierarchical Clustering**: bouwt clusters geleidelijk aan op door de dichtstbijzijndste traces bij elkaar te nemen tot een grotere cluster.
- **Self-organizing Map**: is een neural network techniek die gelijkaardige cases toewijst aan dezelfde of naburige nodes.

Het voordeel van deze benadering is dat deze heel flexibel is. De analist heeft de keuze in welke profiles en dus welke standpunten hij gebruikt, kan vervolgens ook een afstandsmaatstaf naar wens kiezen en heeft tot slot de mogelijkheid om eender welk clusteralgoritme dat kan werken met een afstandsmatrix te gebruiken. Het nadeel is dan weer dat er heel veel onbekenden in het algoritme zitten en dat je met andere woorden heel veel dingen, heel veel factoren, zelf moet beslissen. Deze keuzes kunnen echter een grote impact hebben op het eindresultaat waardoor een grondige kennis van het algoritme en de gevolgen van bepaalde keuzes goed begrepen dienen te zijn alvorens men dit algoritme kan toepassen.

5.2.2 Active Trace Clustering - 2013

De Active Trace Clustering techniek (ActiTraC) van De weerdt et al. [71] baseert zich, in tegenstelling tot wat de meeste andere technieken doen, niet op een vector space model, noch definieert het een maatstaf die de gelijkheid tussen twee procesinstanties kwantificeert. De auteurs benadrukken namelijk dat men rekening moet houden met het verschil tussen enerzijds clustering bias en anderzijds evaluation bias. In een traditionele data mining setting, waarvoor de meeste clustering technieken ontworpen zijn, heeft een clustertechniek het uiteindelijke doel om de gelijkheid binnen clusters (intra-cluster) te maximaliseren en de gelijkheid tussen clusters (inter-cluster) te minimaliseren. Dit wordt binnen de context van trace clustering *clustering bias* genoemd. Het process mining perspectief is echter helemaal anders. Bij process mining, en meer bepaald bij process discovery, moeten de clusters ervoor zorgen dat de precision, fitness en complexiteit van procesmodellen verbeterd worden. Dit is dan weer

de *evaluation bias*. De Weerdt et al. [71] benadrukken dus de sterke divergentie tussen de clustering bias en de evaluation bias. Bijgevolg probeert ActiTraC, gegeven het aantal clusters, een optimale verdeling te vinden waar de gecombineerde accuraatheid van de procesmodellen maximaal wordt. Dit trace clustering algoritme voegt dus, zoals het framework van Sectie 5.1 voorstelt in de blok *extended processing capability*, extra evaluatiemaatstaven toe om het clusteralgoritme beter te begeleiden naar zo optimaal mogelijke clusters.

Het algoritme bestaat uit vier belangrijke stappen. De eerste stap is de initiatiestap waar identieke procesinstanties, cases die exact dezelfde trace hebben, gegroepeerd worden in zogenaamde *distinct process instances (dpi)*. Ook de verzameling van (lege) clusters wordt geïnitialiseerd. Daarna volgen de selectiestap (de tweede stap) en de *Look Ahead* stap (de derde stap). Deze twee gebeuren iteratief. Gedurende de tweede stap worden traces iteratief geselecteerd en wordt er gebruik gemaakt van een *selective sampling strategy*. Concreet betekent dit dat een *active learner* op basis van een informatieve maatstaf kiest welke procesinstantie als eerst volgende geselecteerd wordt.

Het Active Trace clustering algoritme voorziet standaard twee selectiemethodes. Een eerste mogelijkheid is om *Frequency-based selective sampling* te kiezen. Bij deze selectiemethode worden de verschillende dpi's geranschikt volgens het aantal voorkomens. Een andere mogelijkheid is de *Distance-based selective sampling* selectiemethode. Dit is een meer geavanceerde methode die in principe eender welke afstandsmaatstaf toelaat. Standaard kiezen de auteurs ervoor om de afstandsmaatstaf *Maximal Repeat Alphabet euclidean distance* te gebruiken. Bij deze selectiemethode wordt de mate van gelijkenis van de verschillende dpi's bepaald waarna de dpi's op basis van deze maatstaf gesorteerd worden. Deze selectiemethode volgt het idee dat de meest gelijkaardige traces moeten zoveel mogelijk in één cluster.

De selectiemethode zal een volgorde van dpi's bepalen waarna een procesinstantie gekozen kan worden. De geselecteerde procesinstantie wordt vervolgens toegewezen aan de reeds geselecteerd procesinstanties waarna berekend wordt of het procesmodel van de desbetreffende cluster nog steeds accuraat genoeg is. Als de cluster nog een voldoende hoge *fitness* (accuraatheid) heeft, kan de nieuw geselecteerde procesinstantie definitief toegevoegd worden aan de cluster en kan verder gegaan worden met de selectiestap. Is dit echter niet het geval, dan gaat het algoritme over naar stap 3, *Look Ahead*. In deze stap gaat het algoritme kijken of één van de nog niet geselecteerde procesinstanties past binnen de gemaakte cluster van stap 2. Een procesinstantie wordt in deze fase enkel toegevoegd aan de cluster indien de instantie de fitness (accuraatheid) van het procesmodel niet wijzigt. Procesinstanties die niet aan deze voorwaarde voldoen blijven in de event log. Als al de procesinstanties overlopen zijn, keert het algoritme terug naar stap 2 en probeert het de volgende cluster zo goed mogelijk op te vullen. Tot slot is er nog stap 4: *residual trace resolution*. ActiTraC voorziet twee mogelijkheden om de, na iteraties van stap 2 en 3, overgebleven procesinstanties te verwerken. Een eerste optie is om een aparte cluster te creëren met al deze instanties. Een andere optie is om de overgebleven procesinstanties te verdelen over de gecreëerde clusters op basis van de individuele trace fitness voor de verschillende procesmodellen [71].

Analisten die deze techniek willen implementeren dienen goed na te denken over enerzijds het aantal clusters en anderzijds de keuze van de clusterkwaliteitcriteria. Deze twee parameters beïnvoeden namelijk aanzienlijk de kwaliteit van de uiteindelijke procesmodellen.

5.2.3 Markovmodellen - 2015

De meeste trace clustering technieken zijn gelimiteerd tot het vinden en groeperen van structureel gelijkaardige cases. De context van het proces, in het geval van event en case data in de vorm van bijkomende attributen, wordt vaak niet mee in rekening genomen. In [30] wordt een clustering techniek voorgesteld die cases gaat groeperen die gelijkaardig gedrag vertonen op basis van een aantal geselecteerde perspectieven. Dit alles met het Markov cluster algoritme (MCL) als onderliggende basis. MCL gebruikt stochastische matrices die de overgangskansen tussen knooppunten in een *graph* weergeven. Deze stochastische matrices worden vervolgens afwisselend bewerkt met wat men noemt de expansiestap en de inflatiestap. Men verhoogt de waarden eerste met een gegeven macht en normaliseert vervolgens de matrix zodat het opnieuw een stochastische matrix wordt. Door deze stappen te itereren resulteert de matrix in een *graph* met verschillende componenten die vervolgens geïnterpreteerd worden als clusters. Het idee hierachter is dat *random walk*s, startende van een knooppunt, regelmatig gelijkaardige knooppunten zal tegenkomen op zijn weg. De stochastische inputmatrix die gebruikt wordt bij trace clustering is een similarity matrix die weergeeft hoe dicht verschillende cases of traces bij elkaar liggen. Naast gebruik te maken van contextinformatie onderscheidt deze trace clusteringtechniek zich ook in het aantal clusters. Waar veel trace clustering technieken het aantal clusters zien als een inputparameter die bepaalt wordt door de analist, bepaalt dit algoritme zelf, op basis van de data in de event log, hoeveel clusters er uiteindelijk moeten zijn [30]. Dit algoritme werkt dus op basis van drie zelf in te stellen inputparameters: de similarity matrix die op meerdere manieren berekend kan worden, de expansieparameter en tot slot nog de inflatieparameter.

5.2.4 Sequence alignment - 2015

Deze trace clustering techniek gebaseerd op sequence aligment is geïnspireerd door technieken die voornamelijk gebruikt worden in de bio-informatica [27]. De eerste van de vier stappen van deze techniek is het voorbereiden van de event log. Concreet betekent dit hier dat duplicate traces verwijderd worden. Zo wordt de grootte van het clusterprobleem enorm gereduceerd, maar gaat er langs de andere kant ook informatie verloren zoals de frequentie van de traces. In de tweede stap gaat men sequences, een bepaalde opeenvolging van verschillende activiteiten, uitlijnen (aligment) ten opzichte van elkaar en deze een score geven op basis van een scoresysteem. Een simpel scoresysteem kan een score van +1 toewijzen aan exacte overeenkomsten en een score van -1 toewijzen aan mismatches en zogenoemde gaps. Evermann et al. gebruiken in hun werk het *Smith-Waterman-Gotoh (SWG)* aligment algoritme. Dit algoritme heeft twee resulatateN als output: het aantal exacte matches tussen de aligned sequences en de aligment kost in elke sequence. Deze laatste kan gezien worden als een simmilarity matrix en kan dus als input dienen voor de daadwerkelijke clusteralgoritmen [27].

5.2.5 Multi-objective - 2016

De Koninck et al. [19] argumenteren in hun paper dat het clusteren van traces gepaard gaat met verschillende doelstellingen. Zo hebben andere trace clustering technieken doorgaans gefocust op één van de volgende doelstellingen: het optimaliseren van de kwaliteit van het procesmodel, de gelijkheid tussen traces, de mate waarin de clusters overeenkomen met verwachtingen van een domeinexpert of de stabiliteit van het algoritme. De Koninck et al. [19] proberen om verschillende van deze doelstellingen te combineren in een *multi-objective* benadering. De belangrijkste vraag is vervolgens hoe men meerdere doelstellingen kan combineren tot iets betekenisvol. Klassiek zijn er drie opties. Een eerste optie is om een bepaalde hierarchie te creëren tussen de verschillende doelstellingen. Men gaat er hier dus van uit dat de ene doelstelling belangrijker is dan een andere doelstelling. Vervolgens

gaat men zoeken voor Pareto-optimale oplossingen. Een andere mogelijkheid is een gewogen doelfunctie. Hierbij krijgen de verschillende doelstellingen elk een gewicht naargelang hun respectievelijk belang waarna deze samengestelde doelfunctie gezien kan worden als één doelstelling die geoptimaliseerd dient te worden. Een derde mogelijke oplossing is *co-clustering*. Bij co-clustering wordt een clusteroplossing gecreëerd voor elke doelstelling afzonderlijk. Vervolgens worden de verschillende clusteroplossingen gecombineerd tot één algemene oplossing [19].

5.2.6 Compound Trace clustering - 2017

Compound Trace clustering is relatief uniek in die zin dat het algoritme de kwaliteit en de complexiteit van het uiteindelijke model van het subproces mee in rekening neemt tijdens het clusteren van de event log [64]. Het compound trace clustering algoritme bestaat uit twee grote stappen. De eerste stap bevat de trace clustering zelf terwijl de tweede stap gaat proberen om de accuraatheid van de verschillende procesmodellen te verbeteren. De eerste stap werkt volgens het principe van *top-down trace clustering* (TDTC). In plaats van op zoek te gaan naar traces die op elkaar gelijken en het clusteren dus af te laten hangen van een bepaalde maatstaf, zoals afstand, die gelijkheid uitdrukt, gaat men hier proberen om meteen te focussen op de complexiteit van de uiteindelijke procesmodellen. Om dit te bereiken hanteert TDTC een *greedy approach* die gekenmerkt wordt door een optimale gewogen gemiddelde met betrekking tot de complexiteit. Neem bijvoorbeeld een event log *L*. Gegeven deze event log *L* en het aantal uiteindelijke clusters (in dit voorbeeld drie), zal TDTC eerst een optimale manier zoeken om L te verdelen in twee sublogs L1 en L2. Het algoritme doet dit aan de hand van *complexity-related significant trace behaviours*. Een *trace behaviour* wordt hier gedefinieerd als een opeenvolging van activiteiten afkomstig van de event log. Het idee achter het achterhalen van deze *trace behaviours* is dat gelijkaardige sequenties kunnen wijzen op bepaalde subprocessen en waar deze subprocessen dan van elkaar verschillen. Vervolgens kiest het algoritme de sublog met de grootste verwachtte complexiteit en gaat opnieuw zoeken naar een optimale split van L2. Deze sublogs zijn de uiteindelijke output van de eerste stap. In de tweede stap wordt de accuraatheid van de verschillende modellen van de subprocessen verbeterd. Deze stap wordt uitgevoerd met behulp van het algoritme HIF, wat staat voor *a Heuristic method for Improving the Fitness of mined business process models* [65]. HIF gaat zoeken naar gedragspatronen in de event log die niet opgenomen worden door het process discovery algoritme. Het HIF-algoritme gaat vervolgens proberen om deze patronen om te vormen naar structuren die wel uitgedrukt kunnen worden in een procesmodel [64].

5.2.7 A non-compensatory approach - 2017

De *non-compensatory approach* van Delias et al. [21] is een trace clustering techniek die de event log gaat onderverdelen in verschillende sublogs op basis van verschillende criteria. De auteurs vinden immers dat traces clusteren op basis van één enkel criterium equivalent is aan het bewust negeren van bepaalde aspecten van de werkelijkheid [21]. Daarom achtten zij het noodzakelijk om een techniek te ontwikkelen die gebaseerd is op verschillende gelijkheidscriteria die elk hun oorspronkelijke betekenis kunnen behouden. Het onderscheidende van deze paper is daarom ook dat het analisten toelaat om een event log te clusteren op basis van meerdere, naadloos geïntegreerde criteria volgens een *non-compensatory* benadering gebaseerd op de *outranking relations theory*.

Ook deze techniek kan opgedeeld worden in drie delen. Eerst is het belangrijk om de juiste criteria te definiëren. Bij het definiëren van de criteria door de analist is het belangrijk om er op te letten dat de verzameling van deze *n* criteria een consistente familie vormen. Enkele mogelijke criteria binnen process mining zijn bijvoorbeeld: gemeenschappelijke activiteiten tussen twee traces, gemeenschap-

pelijke transities, case attributen of prestatiemaatstaven zoals duurtijd. In de tweede stap worden vervolgens verschillende thresholds bepaald. Aan elk criterium worden drie thresholds toegewezen: de *veto threshold*, de *indifference threshold* en de *similarity threshold*. De *veto threshold* drukt de kracht uit, toegeschreven aan een bepaald criterium, om tegen de bewering van gelijkheid in te gaan in het geval dat twee objecten, traces in dit geval, significant van elkaar verschillen. De *indifference threshold* daarentegen geeft het algoritme de mogelijkheid om aan te geven dat twee objecten niet echt gelijkenissen vertonen maar evenmin significante verschillen. De *similarity threshold*, tot slot, geeft aan dat twee objecten wel grote gelijkenissen vertonen. Naast deze thresholds wordt er voor elk criterium ook nog een gewicht bepaald dat uitdrukt in welke mate dat specifike criterium bijdraagt aan de uiteindelijke gelijkheid tussen twee traces die bepaalt wordt in stap drie. Daarna wordt in de tweede stap de partiële gelijkheidsindex bepaalt om *concordance* of *discordance* tussen twee traces te bepalen en dat voor elk criterium apart voor elke combinatie van twee traces. Deze partiële gelijkheidsindexen weerspiegelen de positieve (*concordant*) of negatieve (*discordant*) bijdrage tot de algemene gelijkheidsindex die in de volgende stap berekend wordt. In de derde stap worden alle partiële gelijkheidsindexen gecombineerd met behulp van de eerder bepaalde gewichten in een *overall concordance index* en een *overall disconcordance index*. Eenmaal deze indexen berekent zijn voor alle combinaties tussen de traces kan men overgaan tot het berekenen van de *overall similarity metric*. Deze gelijkheidsmatrix kan vervolgens gebruikt worden als input voor één van de gekende clustering technieken zoals agglomerative hierachical clustering om de uiteindelijke clusters van traces te bekomen [21].

De keuzevrijheid met betrekking tot de criteria, de thresholds en de gewichten is meteen ook het zwakste punt van deze techniek. De uiteindelijk gedefinieerde gelijkheid tussen traces, die een aggregatie is van de verschillende criteria op basis van de thresholds en de gewichten, is namelijk alles bepalend voor het uiteindelijke clusterresultaat.

5.2.8 Improving the non-compensatory approach - 2021

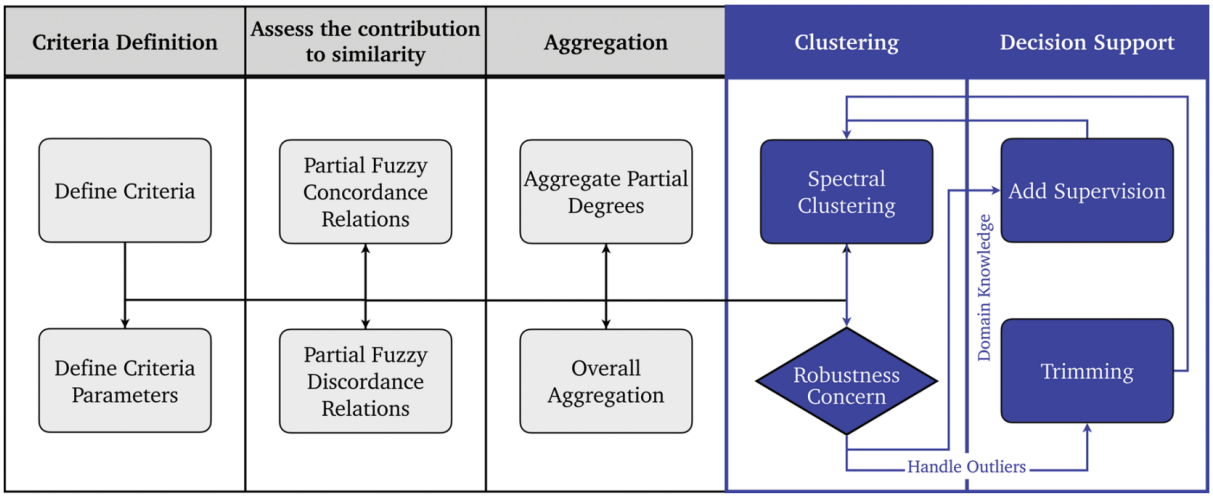
In 2021 kwamen de auteurs van de non-compensatory approach, Delias et al., met een verbetering van hun *non-compensatory approach*[22]. Dit nieuwe algoritme breidt het oorspronkelijke algoritme uit op drie punten. Als eerste stellen ze een verbeterde cluster techniek voor, of beter gezegd een andere benadering. In plaats van hierachial clustering te gebruiken, stellen ze voor om over te schakelen naar spectral clustering omdat deze benadering meer flexibiliteit toe laat. Zo kan het agglomerative hierarchical clustering algoritme bijvoorbeeld enkel convexe verzameling ontdekken. Ook om eventuele uitschieters te behandelen biedt spectral clustering een betere uitgangspositie. Spectral clustering bouwt evenzeer eerst een similarity matrix voor alle te clusteren objecten. Vervolgens zal het algoritme een *spectral embedding* van de objecten uitvoeren wat ervoor zal zorgen dat de clusters beter zichtbaar zijn. Tot slot wordt een traditioneel clusteralgoritme, zoals K-means clustering, gebruikt om de objecten te verdelen over de clusters [22].

Daarnaast introduceren ze *reinforced* en *counterveto* effecten en *pairwise* constraints om het cluster algoritme beter te leiden en om domeinkennis te kunnen integreren in het gehele algoritme. Deze *pairwise* constraints geven concrete relaties aan tussen twee traces die aangegeven worden door een procesexpert; meer bepaald of twee traces al dan niet tot dezelfde cluster zouden moeten behoren. Deze constraints worden in de praktijk gebracht aan de hand van twee nieuwe thresholds: een *reinforced preference threshold* en een *counterveto threshold*. Deze thresholds zullen dienen als aanbevelingen voor het algoritme om bepaalde objecten best samen in één cluster te zetten (*reinforced preference*) of net niet samen in één cluster te zetten (*counterveto*) [22].

Tot slot bedachten de auteurs ook een manier om uitschieters te behandelen met behulp van een

trimming benadering zoals een integer lineair programmeringsprobleem. Uitschieters hebben namelijk een groot effect op de homogeniteit van de clusters [22]. Deze *trimming* benadering verwijderd de uitschieters en clustert vervolgens de overblijvende objecten. Of een object al dan niet een uitschieter wordt, wordt bepaald door te kijken hoe dicht een bepaald object staat bij een andere verzameling van objecten. De volgende vraag die zich stelt is dan: vanaf welke afstand kan een object als een uitschieter beschouwd worden? Om deze vraag op te lossen hebben de auteurs gekozen om dit probleem te omschrijven als een lineair programmeringsprobleem waarbij het aantal te verwijderen uitschieters, *k*, gezien wordt als een op voorhand in te stellen parameter [22].

Het gehele clusterproces van dit trace clustering algoritme wordt door de auteurs voorgesteld aan de hand van een afbeelding, zie Figuur 9 [22], die duidelijk is opgesplitst in de verschillende fases van het algoritme. Zo worden eerst de criteria gedefinieerd waarna vervolgens de bijdrage van deze criteria tot de algemene gelijkheidsindex wordt berekend. Deze bijdrage dient dan als basis voor de aggregatie die als volgt weer dient als input van het clusteralgoritme waar domeinkennis een onderdeel van is.



Figuur 9: De verschillende stappen van de verbeterde versie van het non-compensatory trace clustering algoritme [22]. De blauwe delen zijn de specifieke bijdrage van de verbetering.

5.2.9 Expert-driven - 2021

De huidige trace clustering technieken zijn doorgaans gebaseerd op een gelijkheidsmaatstaf tussen verschillende traces of worden aangestuurd door de kwaliteit van het procesmodel van de desbetreffende cluster. Dergelijke technieken zijn echter moeilijk te evalueren of te verantwoorden door domeinexperten. De Koninck et al. [20] hebben daarom een nieuwe trace clusteringtechniek ontwikkeld die het mogelijk maakt om bepaalde kennis van experten mee te nemen bij het clusteren van traces. De auteurs van [20] lossen dit probleem op door twee types van constraints mee in rekening te nemen bij het clusterproces: de *must-link constraints* en de *cannot-link constraints*. Ze stappen hiermee dus het pad van *constrained clustering* op [70]. De basisprincipes van *constrained trace clustering* zijn eigenlijk eenvoudig. Bepaalde *instances* worden gedwongen bij elkaar genomen in één cluster terwijl andere *instances* net doelbewust van elkaar gescheiden worden [70]. Echter is het belangrijk om de kwaliteit van de constraints goed in het oog te houden. Het al dan niet opnemen van bepaalde constraints kan namelijk een negatieve impact hebben op het uiteindelijke resultaat [20]. Bij het opstellen van de constraints is het belangrijk dat er gekeken wordt naar de hoeveelheid informatie die vervat zit in de constraints en meer bepaald de hoeveelheid informatie die in de constraints vervat zit die het algoritme niet zelf had kunnen ontdekken. Daarnaast is de coherentie tussen de verschillende cons-

traints ook een belangrijke maatstaf voor de kwaliteit van de constraints. Om op deze maatstaf goed te scoren is het belangrijk dat *must-link constraints* en *cannot-link constraints* elkaar niet tegenwerken [20]. Het correct bepalen van de constraints is naast een voordeel, als er voldoende kennis aanwezig is, ook een nadeel. De constraints dienen eerst en vooral opgemaakt te worden door iemand die een goed overkoepelend beeld heeft van het hele proces. Deze persoon is echter niet altijd aanwezig. Daarnaast kunnen deze constraints het uiteindelijke resultaat ook negatief beïnvloeden doordat een deel van het resultaat al op voorhand vast ligt. Het juist bepalen van dergelijke constraints is dus met andere woorden helemaal niet zo eenvoudig.

In praktrijk kunnen dergelijke constraints op verschillende manieren mee in rekening worden genomen door het algoritme. Zo kunnen bestaande trace clustering algoritmen aangepast worden tot trace clustering technieken die constraints, en dus kennis van experten, wel mee in rekening nemen door het simpel wijzigen van de afstandsmatrix voor het werkelijk clustering algoritme begint. De afstanden dienen zo aangepast te worden zodat de afstanden tussen *instances* met *must-link constraints* zeer klein worden en zeer groot voor *instances* met *cannot-link constraints*. De Koninck et al. [20] ontwikkelden echter ook een nieuwe techniek genaamd ConDriTrac - *Constrained Driven Trace Clustering*. Deze techniek bestaat uit drie fasen waarbij eerst de *traces* gelinkt aan *cannot-link constraints* over de clusters verdeeld worden. In de tweede fase worden vervolgens de andere traces verdeeld over de clusters waar deze het best passen. Deze beslissing wordt door het algoritme genomen op basis van het procesmodel van elke cluster. En tot slot worden de overgebleven traces in de derde fase toegewezen aan ofwel een nieuwe cluster ofwel de best mogelijke cluster.

Om een overkoepelend beeld te krijgen van de opdeling van de verschillende trace clustering technieken, is het van groot belang eerst een goed beeld te krijgen van de verschillende aspecten die de verschillende technieken karakteriseren [67]. Zo wordt een clustertechniek doorgaans ontwikkeld met een concreet doel voor ogen zoals in de eerste paragraaf van deze sectie kort werd aangehaald. Daarnaast wordt ook algemeen aangenomen dat de trace representation een niet te miskennen impact heeft op de uiteindelijke clusterresultaten [67, 33]. Daarnaast zijn er ook nog theoretische aspecten van clustermethoden die de algemene literatuur in acht neemt. Hiertoe worden vooral de keuze van de afstandsmaatstaf en de clusterbenadering gerekend [67, 33].

6 Trace representation

In de vorige sectie werden verschillende trace clustering technieken besproken. Bij de verschillende trace clustering technieken werd er vaak vanuit gegaan dat de verschillende traces die aanwezig zijn in een event log maar op één manier gerepresenteerd kunnen worden. Echter gaat men er in de literatuur van uit dat de trace representatie, de manier waarop een trace weergegeven wordt en als input dient voor het clustering algoritme, een belangrijke impact heeft op de kwaliteit van het process discovery probleem [11, 18]. In deze sectie wordt eerst besproken waarom het probleem van trace representatie wel degelijk een probleem is. Vervolgens worden verschillende technieken behandelt die voor een betere trace representatie zouden moeten zorgen.

Een event log bestaat doorgaans uit meer attributen dan enkel maar de case-id, de activiteiten, de uitvoerder en een tijdstip. Vaak is de event log uitgebreider en bevat deze ook data-attributen die een invloed hebben op de uitvoering van het proces dat bestudeerd wordt. Zelfs als deze extra data-attributen niet standaard in de event log zitten, is het vaak mogelijk om de event log alsnog uit te breiden door data van bedrijfssystemen toe te voegen. Het representeren van een trace wordt op deze manier een bijna typisch machine learning probleem waarbij de moeilijkheid is om een multidimensionale vector om te vormen naar een *low dimensional* vector zonder veel betekenisvolle informatie te verliezen [11, 18].

6.1 Klassieke trace representation technieken

Er zijn enkele klassieke trace representation technieken die doorgaans gebruikt worden bij trace clustering technieken. In de volgende alinea's worden enkele veelgebruikte trace representation technieken besproken:

- Bag of activities
- K-gram
- Maximal repeat
- Distance graph

6.1.1 Bag of activities

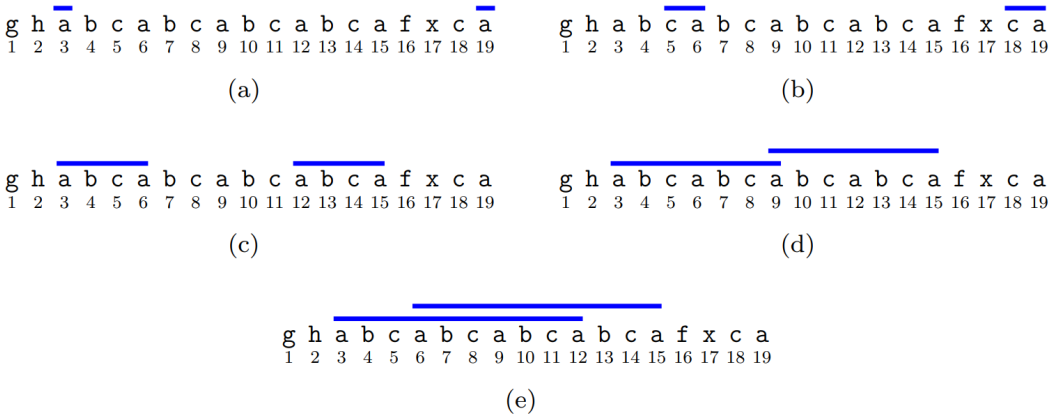
Bag of activities is een eerste techniek die gebruikt wordt bij het representeren van een trace. Bij deze techniek transformeert men een trace simpelweg in een vector van verschillende activiteiten die voorkomen in de event log. Enkel de informatie met betrekking tot welke activiteiten voorkomen en welke niet wordt dus opgenomen in deze vector. Elke trace wordt geconverteerd in zo een vector in de vorm van een *binary vector space model*. Hierbij wordt het overeenstemmende element in de vector op 1 gezet indien de activiteit voorkomt in de trace, en op 0 in het andere geval [11]. Een andere mogelijkheid is om af te stappen van de binary vector space model en de 1 en 0 te vervangen door het aantal keren dat een activiteit in de trace voorkomt [60]. Aan deze techniek, bag of activities, zijn twee grote nadelen verbonden. Ten eerste geeft het simpelweg aangeven of een activiteit al dan niet voorkomt in een trace geen contextinformatie mee en net deze informatie is cruciaal om een trace beter te plaatsen in de uitvoering van een proces. Ten tweede zegt deze methode ook niets over de volgorde van uitvoering van de verschillende activiteiten waardoor het control-flow perspectief naar de achtergrond verdwijnt. Ten derde gaat, afhankelijk van de gebruikte variant, ook nog informatie verloren met betrekking tot het aantal keer dat een activiteit voorkomt binnen één trace [60].

6.1.2 K-grams

Een tweede vaak voorkomende techniek is *K-grams*. Een K-gram model probeert om de context, i.e. control-flow informatie, van het proces mee in rekening te nemen in die zin dat het K-gram model rekening houdt met subsequences van activiteiten. Het voordeel hierbij is dat deze subsequences de volgorde van uitvoering van de verschillende activiteiten wel in rekening neemt [60]. Echter moet de opmerking gemaakt worden dat de context van een proces meer is dan enkel de opeenvolging van de activiteiten. Men spreekt in de literatuur van K-grams om aan te geven dat men spreekt over een opeenvolging van *k* activiteiten. Een voorbeeld om dit principe even te verduidelijken: voor de trace *abacaab*, is de verzameling van 2-grams gelijk aan {ab, ba, ac, ca, aa}. De verzameling van 3-grams is dan weer gelijk aan {aba, bac, aca, caa, aab} [60].

6.1.3 Maximal Repeat

Een derde klassieke techniek is de techniek genaamd *Maximal Repeat*. In [32] definieert de auteur maximal repeat als volgt: een maximaal paar in een reeks *s* is een subreeks α die zich afspeelt in *s* op twee verschillende posities *i* en *j* zo dat het element direct links (rechts) van de subreeks α op positie i verschillend is van het element aan de linkerzijde (rechterzijde) van de subreeks α op positie j. In [32] wordt de trace $t_1 = ghabcabcabcabcafxca$ als voorbeeld gegeven. In Figuur 10 worden voor deze trace de *maximal pairs* aangeduid. Bijgevolg zijn de maximal repeats voor deze trace {a, ca, abca, abcabca, abcabcabca}.



Figuur 10: De verschillende *maximal pairs* die voortkomen uit de trace $t_1 = ghabcabcabcabcafxca$. De verzameling van maximal repeats is dan als volgt: {a, ca, abca, abcabca, abcabcabca}.

6.1.4 Distance graph theory

Tot slot is er ook nog de *Distance graph theory*: gegeven een corpus C, is de distance graph met orde *k* van een document D gegenereerd van C gedefinieerd als *graph* G(C; D; k) = (N(C); A(D; k)), waar N(C) gelijk is aan de verzameling van nodes, of met andere woorden gelijk is aan the verzameling van unieke woorden in de gehele corpus C, en waarbij A(D; k) gelijk is aan de verzameling van edges in de *graph*. De verzameling A(D; k) bevat de verzameling van *directed edges* van node *i* naar node *j* als het woord *i* voorafgaat aan het woord *j* door minimaal *k* posities [11]. Deze theorie kan ook toegepast worden op het probleem van trace representation. Hierbij kan de verzameling van activiteiten A gezien worden als de verzameling van de zogenoemde unieke woorden in corpus C, en kan een trace gezien worden als een document D. Door de event log met de bijhorende traces op deze manier te bekijken kan een distance graph geconstrueerd worden.

Het grote nadeel van bovengenoemde representatiemethoden is dat de dimensie van de vector space spectucalair stijgt voor real-life event logs tot tienduizenden. Dit heeft een niet te miskennen impact op de prestatie van de verschillende clusteralgoritmen die gebruik maken van deze representatiemethoden. Met dit in het achterhoofd en met de eerder aangekaarte challenge van lage dimensionaliteit met veel informatieve waarde is de uitdaging dus om een representatiemethode te vinden die het aantal dimensies reduceert.

6.2 Geavanceerde trace representatietechnieken

Naast de standaard representatietechnieken die doorgaans gebruikt worden, is er ook gezocht naar meer geavanceerde technieken om traces op een betekenisvolle manier te representeren. Het basisidee hier achter is steeds het concept van deep neural networks.

Deep Neural Networks zijn een verzameling van machine learning algoritmen die computers toelaten om te leren op verschillende abstractieniveaus [11]. Eén van de doelen van deep neural networks is om de inputwaarde X te transformeren naar een compacte tussentijdse representatie zodat deze accuraat de outputwaarde kan voorspellen [11].

Het basisidee is om een menselijk brein te imiteren. Dit wordt typisch gedaan door middel van verschillende zogenoemde *layers* die zich bevinden tussen de input en de output *layers*. Deze layers worden ook wel hidden layers genoemd. Deze hidden layers bestaan uit een groot aantal *neuronen* die aan elkaar gekoppeld worden om de informatie beter te verwerken. De neuronen worden aan elkaar gekoppeld door middel van activatiefuncties en gewichten. In de volgende subsecties worden verschillende meer geavanceerde trace representation technieken gebaseerd op deep neural networks nader besproken.

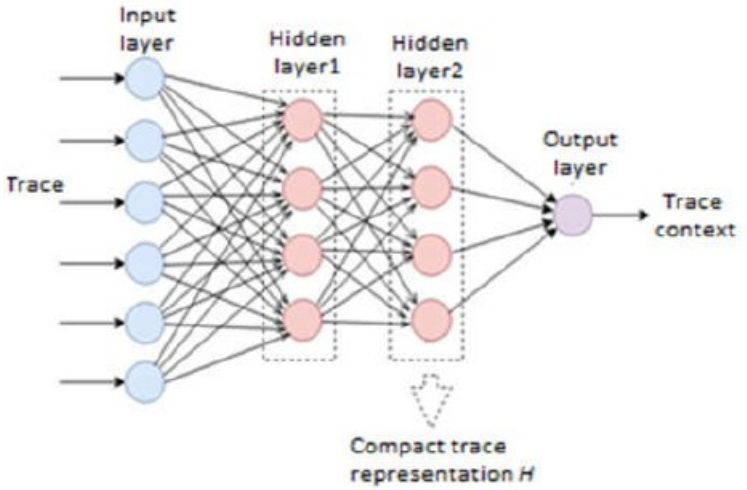
6.2.1 A compact trace representation

In Bui et al. [11] beschrijven de verschillende auteurs een techniek die het mogelijk maakt om met behulp van DNN een trace compact voor te stellen zonder noodzakelijke informatie te verliezen. De methode van Bui et al. bestaat uit drie grote stappen waarbij ze de techniek van supervised learning gebruiken. Eerst wordt de outputwaarde Y berekend. Vervolgens wordt de output vergeleken met de gewenste waarde Z. De derde stap is afhankelijk van het resultaat van de tweede stap. Als de gewenste waarde Z niet gelijk is aan de outputwaarde Y, dan worden de gewichten van het deep neural network en de bias aangepast waarna de ouputwaarde Y opnieuw wordt berekend.

Deze techniek kan ook toegepast worden om een trace op een betekenisvolle manier weer te geven. Hierbij wordt de verzameling van traces in de event log beschouwd als de input data, en de trace context wordt beschouwd als de gelabelde data Z. Het deep neural network bestaat uit één inputlaag *T*, 2 hidden layers *H* en één outputlaag *O*. Deze opbouw wordt weergegeven in Figuur 11. Eén van de doelen van deep neural networks is om de inputwaarde X, in dit geval een klassieke trace representation, om te vormen naar een compact tussentijdse representatie, in dit geval de hidden layer *H*. Deze hidden layer geeft de informatie die verborgen zit in de input op een betere en meer compacte manier weer die het mogelijk maakt om de outputwaarde te voorspellen. Dus in plaats van een klassieke trace representation methode te gebruiken, wordt hier geopteerd om een compacte, meer informatieve methode te gebruiken [11].

Neem een aanvraag voor een krediet bij de bank als voorbeeld. Deze aanvraag heeft verschillende procedures naargelang het type van het krediet. Zo zal een klant van de bank andere stappen moeten ondernemen voor een bedrijfslening dan voor een lening bestemd voor een huis. Elke procedure

heeft een aantal specifieke karakteristieken of activiteiten die verschillen van een andere procedure. Deze karakteristieken en activiteiten worden gedefinieerd als de trace context. Om de techniek van supervised learning met behulp van deep learning networks te kunnen toepassen moeten bepaalde process mining begrippen gelinkt worden aan begrippen gelinkt traditionele deep learning networks technieken. Zo wordt de verzameling van traces gezien als de verzameling van input data, en wordt de trace context als gelabelde data Z beschouwd. Elke input-neuron krijgt een trace *t* toegewezen die wordt gerepresenteerd door middel van een klassieke trace representation techniek zoals de techniek *binary bag of activities*. Deze inputdata wordt in de hidden layers omgevormd door middel van de gewichten en de activatiefuncties. Deze activatiefuncties zijn een in te stellen parameter en de specifieke gewichten worden bepaald door het trainen van het deep neural network. De laatste hidden layer, in Figuur 11 is dat hidden layer 2, wordt, na het trainen van het deep neural network, gebruikt als compacte trace representation.



Figuur 11: Een mogelijke uitwerking van een deep neural network toegepast op trace representation [11].

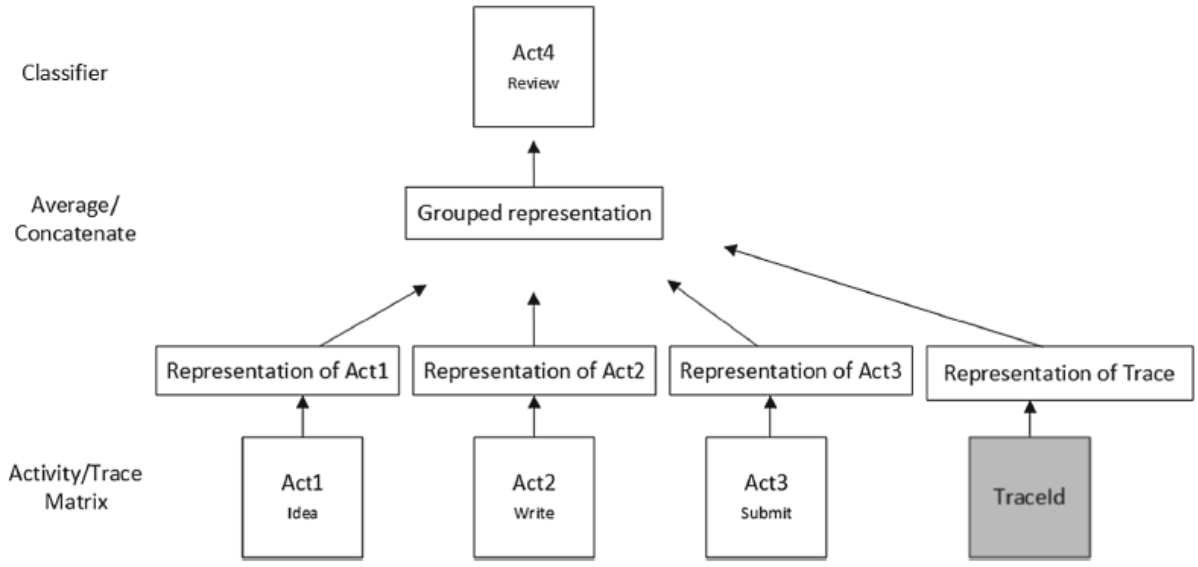
6.2.2 De Trace2Vec-methode

NLP Een mogelijke andere oplossing is om bepaalde technieken van andere disciplines, waar ze met een gelijkaardig probleem zitten, over te nemen. Andere disciplines waar ze eveneens het dimensionaliteitsprobleem hebben zijn: *natural language processing* (NLP), *image recognition* en *social network analytics* [18]. Zo is *representation learning*, waar men zich bezig houdt met de vraag hoe men gegevens op een desbetreffende manier kan weergeven zodat algoritmen hier makkelijk mee aan de slag kunnen, oorspronkelijk ontwikkeld binnen het domein van NLP, maar zijn deze technieken ook bijzonder nuttig daarbuiten.

Representation learning technieken steunen voornamelijk op neurale netwerken om vectoren te maken die bepaalde objecten representeren. Waar objecten binnen het domein van NLP verwijzen naar woorden, zinnen of hele documenten is dit binnen het gebied van business process management lichtjes anders. Woorden worden bijvoorbeeld activiteiten en zinnen worden bijvoorbeeld traces. In [18] worden verschillende technieken uit het domein van NLP aangepast zodat deze ook nuttig zijn voor het vakgebied process mining. Zo is *trace2vec* een methode gebaseerd op de *doc2vec* benadering. Het idee achter *doc2vec* wordt het *Distributed Model of Paragraph Vectors (PV-DM)* [39] genoemd en is een lichte aanpassing van de *Continuous Bag of Words (CBOW)* architectuur ontwikkelt door Mikolov et al. [49]. De CBOW architectuur is ontwikkeld om grote hoeveelheden woorden op een efficiënte manier te ver-

werken en het probeert aan de hand van een neural network, dat een input layer, een projection layer en output layer bevat, het huidige woord te voorspellen op basis van de context. Deze context wordt hier gedefinieerd als de omringende woorden van het desbetreffende woord [49]. PV-DM verschilt van CBOW in die zin dat het meer contextinformatie meeneemt in de uiteindelijke output. In het *doc2vec*-algoritme vertaalt zich dat dan in het meenemen van een paragraaftoken die het neural network de mogelijkheid geeft om meer informatie mee in rekening te nemen bij de uiteindelijke voorspelling en die kan dienen als een geheugen van het onderwerp van de paragraaf [39].

Het idee achter deze methode kan ook vertaald worden naar het representeren van traces. In Figuur 12 wordt het neuraal netwerk visueel voorgesteld. Doordat de trace aan de hand van een *trace identifier* mee wordt genomen in het neural network, wordt het mogelijk om de representaties van activiteiten en traces gezamenlijk mee te nemen in het leerproces van het algoritme. Nu, het algoritme is opgesteld om een bepaalde activiteit te voorspellen gegeven een bepaalde context. Voor toepasbaar te zijn in het trace clustering probleem wordt er op zoek gegaan naar een trace representatie die met zo weinig mogelijk dimensies, een trace zo goed als mogelijk representeert. Om de *trace2vec* methode dus toe te passen in het trace clustering probleem is er een tussenstap nodig. Als men *trace2vec* wilt gebruiken bij trace clustering zal het neural network eerst getraind moeten worden om op een zo goed mogelijke manier activiteiten te voorspellen. Hiervoor kan men onder andere de window size en de lengte van de feature vector aanpassen. Eens het netwerk getraind is, kunnen de trace representaties die verscholen zitten in de hidden layer gebruikt worden voor een cluster algoritme.

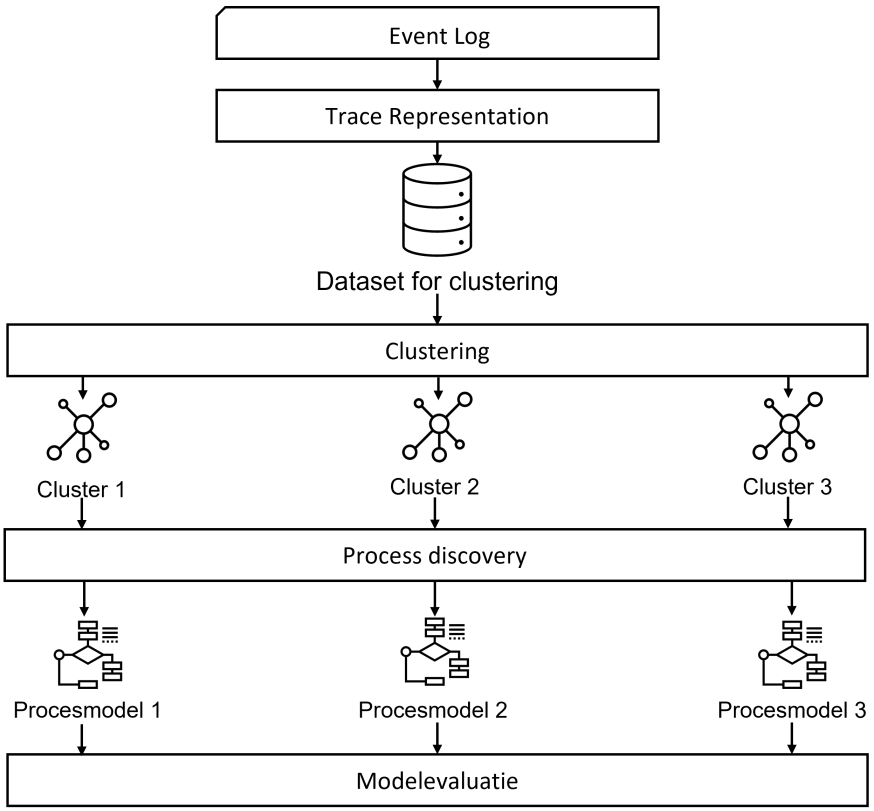


Figuur 12: De *trace2vec*-architectuur met als doel het leren van vectorrepresentaties van traces [18].

7 Methodologie van de vergelijkende analyse

De laatste jaren zijn er tal van verschillende trace clustering algoritmen ontwikkeld met vaak verschillende invalshoeken. Dat bewijst de bespreking van de verschillende technieken in Sectie 5. Echter is er geen uitgebreide vergelijkende analyse van de verschillende trace clustering technieken om te achterhalen welke methode het beste is. Of er überhaupt een beste methode bestaat. En welke methoden in welke situaties het beste presteert. In deze sectie wordt een poging gedaan om dit uit te klaren aan de hand van een vergelijkende analyse van de verschillende trace clustering technieken. Eerst wordt de methodologie van deze analyse toegelicht. Daarna worden de resultaten besproken in de volgende sectie.

De werkelijke analyse wordt opgedeeld in twee grote delen. Eerst wordt er een exploratieve analyse gedaan van de event logs die een eerste beeld moet schetsen van het geobserveerde proces. Daarnaast wordt een vergelijkende analyse van verschillende trace clustering technieken uitgevoerd. De vergelijkende analyse is gebaseerd op het *three-phase framework of process discovery* zoals te zien is in Figuur 13 [28]. Dit framework beschrijft het algemene proces van process discovery gebruik makende van trace clustering. In dit framework kunnen duidelijk drie stappen onderscheiden worden. Een eerste stap is het clusteren van de traces. Deze stap heeft als input een event log en als output verschillende clusters bestaande uit events. De tweede stap is het ontdekken van het procesmodel door een algoritme. De input voor deze stap zijn de verschillende clusters die omgevormd worden tot de output in de vorm van verschillende procesmodellen. Tot slot bestaat de derde stap uit het evalueren van de verschillende procesmodellen aan de hand van evaluatiemaatstaven.



Figuur 13: Het *three-phase framework* betreffende process discovery [28].

7.1 Selectie van trace clustering technieken

Om verschillende trace clustering technieken met elkaar te kunnen vergelijken is het noodzakelijk dat deze technieken praktisch toepasbaar zijn. Bij het doorlopen van de literatuur met betrekking tot trace clustering technieken werd al snel duidelijk dat, als de techniek openbaar ter beschikking werd gesteld, dit meestal was in het kader van het Prom framework [63]. Een eerste selectiecriteria was daarom de beschikbaarheid van een Prom plug-in in ofwel ProM 5 ofwel ProM 6.11 [69]. Naast de beschikbaarheid van een plug-in moet de plug-in ook een exportfunctie hebben. Dit om te verzekeren dat de vergelijkende analyse op een degelijke manier uitgevoerd kan worden. Na een grondige zoektocht, bleken enkel de volgende technieken een werkende plug-in te hebben:

Active Trace Clustering van Deweerdt et al. [71] als plug-in voor ProM 6.11 met meerdere configuratieparameters. Specifiek is er in deze paper gekozen om de trace selectie methode, de target fitness en het maximaal aantal clusters telkens te veranderen. Voor de trace selectiemethode worden beide opties met elkaar vergeleken. Hierbij worden de volgorde van de traces bepaald op basis van de frequentie ofwel op basis van MRA. Voor de target fitness worden er twee waarden gekozen. Omdat het ideale procesmodel, uitgaande van goede kwaliteitsmaatstaven, een fitness van 1 heeft, wordt in één setting voor een target fitness van 1 gekozen. In een andere configuratie wordt een target fitness van 0,8 ingesteld om het algoritme wat meer vrijheid te geven. Tot slot is er bij alle clustering technieken voor gekozen voor drie of zes clusters.

Profiles Het uitgebreide algoritme gebaseerd op profiles van Song et al. [62] als plug-in voor ProM 5. Deze plug-in kent verschillende configuratieparameters. Concreet kan de gebruiker de afstandsmaatstaf en het clusteralgoritme kiezen en heeft de gebruiker ook nog keuze in verschillende profiles die gecombineerd kunnen worden. Voor deze studie is er gekozen om te werken met de afstandsmaatstaf *Euclidean distance measure*. Verder worden de clusteralgoritmen *Self-Organizing Maps* (SOM) en *K-means clustering* met elkaar vergeleken. Bij het SOM-algoritme werden ook nog verschillende configuraties van de breedte en de hoogte met elkaar vergeleken. Volgende combinaties komen aan bod in deze studie: 2-3, 1-3, 4-4 en 5-5. Tot slot is er gekozen voor volgende profiles: *activity* (het al dan niet aanwezig zijn van activiteiten in een trace), *activity patterns* (het al dan niet aanwezig zijn van bepaalde patronen van activiteiten in een trace), *data value* (de waarde van de verschillende data attributen; afhankelijk van de event log) en *performance* (de doorlooptijd).

Sequence Alignment van Evermann et al. Bij dit algoritme kan enkel het aantal clusters ingesteld worden [27]. Ook hier wordt gekozen voor drie en zes clusters.

Het valt meteen op dat het aantal plug-ins voor ProM of andere beschikbare tools voor trace clustering relatief beperkt is in aantal. Bovendien zijn verschillende technieken enkel beschikbaar in Prom 5, een verouderd framework. Merk op dat dit een eerste beperking kan zijn om als analist trace clustering te overwegen als deel van een process mining project.

7.2 Beschrijving event logs

Om de verschillende technieken uitgebreid en in reële situaties te kunnen testen is het belangrijk om kwalitatieve event logs te hebben. Met kwalitatieve event logs worden hier event logs bedoeld die zo goed als mogelijk lijken op processen uit de echte wereld. Processen waarbij onverwachte dingen kunnen gebeuren en waar men moet inspelen op de flexibele noden van de klant. Om aan deze vereisten tegemoet te komen, wordt in dit werk gekozen voor real-life event logs die ter beschikking worden gesteld voor bijvoorbeeld de jaarlijkse *Business Process Intelligence Challenge* (BPIC). De event data van al de afgelopen BPIC's samen met extra informatie over het proces kan teruggevonden worden

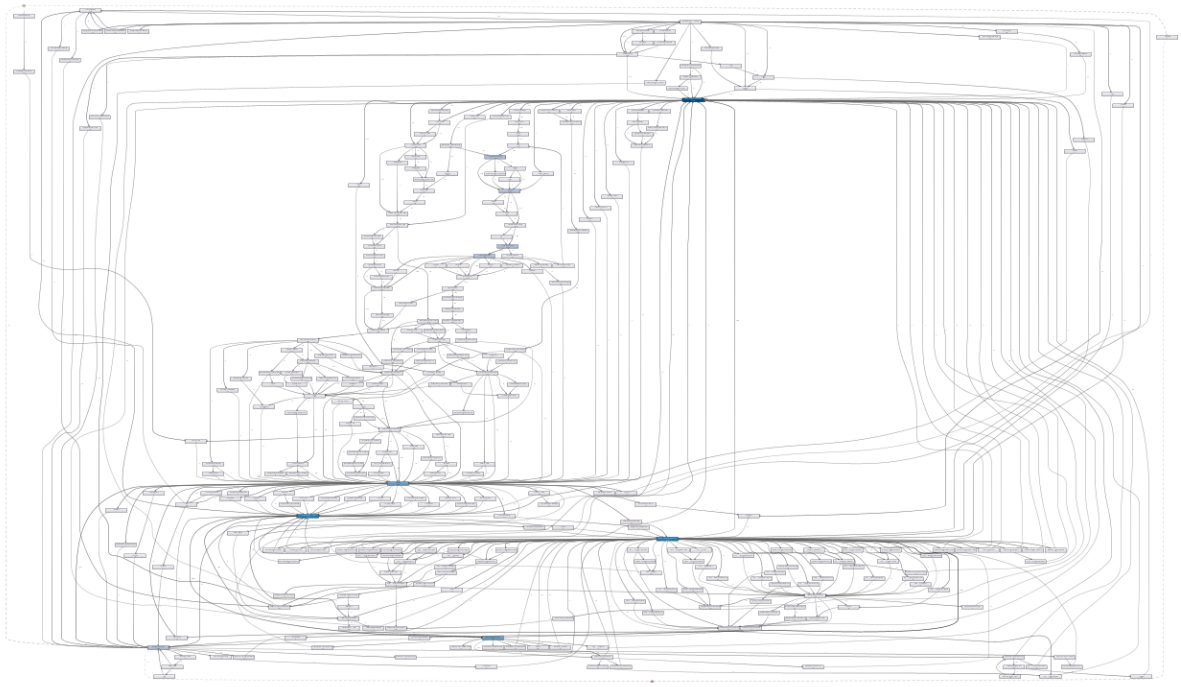
op www.processmining.org [26]. Daarnaast werd ook de database data.4tu.nl doorzocht voor geschikte event logs. Verder in deze subsectie worden de drie geselecteerde event logs toegelicht.

Het eerste proces dat gebruikt zal worden in de vergelijkende analyse is afkomstig uit een ziekenhuis en werd gebruikt in de BPIC van 2011 [24]. De gehele event log bestaat uit 150.000 events verdeeld over meer dan 1100 cases. Met uitzondering van de nodige anonimisatie van de data bevat de event log alle data zoals deze opgeslagen zit in het systeem van het ziekenhuis. Concreet gaat het over de patiënten van de gynaecologische afdeling. Naast de minimum noodzakelijke attributen zoals een case identifier en een activiteit voor elk event, bevat de event log ook verschillende andere, voor het proces relevante, attributen.

De log bevat maar liefst 624 verschillende activiteiten. Door dit hoge aantal activiteiten kunnen de 1143 cases ook verdeeld worden over 981 traces. Dat betekent met andere woorden dus dat bijna elke case een uniek pad aflegt doorheen deze afdeling van het ziekenhuis. Dit maakt de log meteen tot de meest ingewikkelde event log die in deze studie mee in beschouwing wordt genomen. Naast het feit dat er veel unieke traces vervat zitten in de event log, bestaat een trace ook nog eens uit gemiddeld 131,49 events. De originele event log bevat in totaal ook 131 attributen. Om de analyse en de trace clustering overzichtelijk te houden is ervoor gekozen om de belangrijkste attributen hieruit te selecteren. Er is gekozen voor de volgende attributen: CASE concept name, activity id, activity instance id, lifecycle id, resource id, timestamp, case id, case diagnosis, case treatment code en case specialism code. Deze keuze is gemaakt op basis van beperkte beschikbare informatie. Doorslaggevende elementen waren de betekenis van deze attributen die afgeleid kon worden uit de attribuutnaam en het aantal NA waarden per attribuut. Naast een beperkt aantal attributen moet er omwille van beperkte computerkracht ook een selectie gemaakt worden in het aantal cases dat meegenomen wordt in de analyse. Omdat deze event log zo ingewikkeld is, is er op basis van trial and error gekozen voor 275 cases random geselecteerd uit de event log. Dit aantal is tot stand gekomen door het clusteralgoritme SOM op basis van verschillende profiles [62] uit te voeren op verschillende groottes van event logs. Vervolgens werd er een threshold van vier uur gehanteerd om een keuze te maken. Figuur 14 geeft een procesmodel weer afkomstig van Disco waarbij slechts 50% van de traces wordt weergegeven in het model. Deze figuur toont aan dat deze event log onmogelijk op een degelijke manier geanalyseerd kan worden zonder voorgaande preprocessing stappen zoals trace clustering.

Een tweede event log bevat data omtrent het ticketing proces op een help desk van een Italiaans software bedrijf [57]. Deze event log bestaat uit 21 348 events die verdeeld zijn over 14 activiteiten en 4580 cases. In totaal zijn er 226 verschillende traces met gemiddeld 4,66 events per trace. Naast de gebruikelijke attributen die de meeste event logs bevatten zoals een timestamp en een case identifier, bevat deze event log ook nog extra informatie met betrekking tot het proces. Zo krijgt elke case een variant index, een verantwoordelijke, een service level etc. toegewezen en wordt ook elke case beoordeelt op hoe ernstig het probleem is. Deze attributen worden, waar mogelijk, opgenomen in de vergelijkende analyse. Door het beperkt aantal activiteiten en het relatief beperkt aantal traces kan deze log gerekend worden tot de eenvoudigste event log die wordt opgenomen in deze vergelijkende analyse.

Een derde en laatste event log die mee wordt opgenomen, is opnieuw een event log van een *Business Process Intelligence Challenge* (BPIC) [23]. Dit keer afkomstig uit het jaar 2020 en met betrekking op *Travel Permits*. Het proces dat in beschouwing wordt genomen heeft betrekking op toestemmingen in verband met reizen in academisch verband aan een Nederlandse universiteit. De verschillende documenten die nodig zijn om een reistoestemming te krijgen volgen elk een gelijkaardig proces en worden daarom opgenomen in een event log. In totaal bestaat de event log uit 86 581 events die samen deel uit maken van 7065 cases bestaande uit 51 verschillende activiteiten. Al deze cases zijn te verdelen over



Figuur 14: Een spaghetti model afkomstig van de event log van BPIC 2011 [24] (slechts 50% van de activiteiten zit vervat in het model).

1478 traces die een gemiddelde lengte hebben van 12,25 events. De totale dataset telt 176 attributen. Net zoals bij de dataset van de BPI Challenge uit 2011, die hierboven werd besproken, wordt ook hier een keuze gemaakt aan de hand van de betekenis van de attriubten en het aantal NA waarden van een attribuut. De gekozen attributen zijn de volgende: case concept name, activity id, activity instance id, lifecycle id, resource id, timestamp, organisation role, case TaskNumber, case requestedBudget, case BudgetNumber, case declarationNumber 0, case id en tot slot case organizationalEntity 0. Met behulp van deze parameters worden in een volgende fase van de analyse de verschillende clusters bepaald.

In aanloop naar de definitieve keuze van deze drie event logs zijn er eveneens verschillende andere event logs bekeken en vergeleken. In de online open toegankelijke database van de TU Eindhoven (data.4tu.nl) zijn namelijk tal van, steeds geanonimiseerde, real-life event logs beschikbaar. Een eerste, niet te ontlopen, voorwaarde was dat de event log kon ingelezen worden met de beperkte beschikbare computerkracht. Door deze voorwaarde kon bijvoorbeeld de event log van de BPI Challenge 2018 niet weerhouden worden. Daarnaast moeten de verschillende event logs divers genoeg zijn om eventuele effecten op de trace clustering technieken te kunnen achterhalen. Concreet is er in de selectie een keuze gemaakt voor een event log waar het aantal cases ongeveer gelijk is aan het aantal traces en waar er dus met andere woorden een grote diversiteit is. Daarnaast moest de selectie ook een event log bevatten die alsnog een hoog aantal traces heeft maar waarbij dit aantal, mede door het beperkt aantal activiteiten, nog behapbaar blijft. En tot slot een event log die tussen deze twee extremen in blijft en met andere woorden een middenweg vormt. Het gaat dan respectievelijk over de event log van BPIC 2011 Ziekenhuis, de helpdesk en BPIC 2020 Travel Permit.

7.3 Evaluatiemaatstaven

Om de verschillen tussen de trace clustering technieken naar boven te brengen, zijn verschillende evaluatiemaatstaven noodzakelijk. Om de juiste maatstaven te kunnen selecteren is het belangrijk

om het doel van trace clustering voor ogen te houden. Trace clustering technieken worden namelijk ontwikkeld om de kwaliteit van de procesmodellen te verbeteren en dan voornamelijk de precisie en de simpliciteit te verhogen.

Met het doel van trace clustering en de verschillende besproken kwaliteitsmaatstaven van process mining in het achterhoofd zijn de volgende maatstaven geselecteerd:

- **Fitness:** Replay Fitness waarbij gebruik wordt gemaakt van de alignment methode [7]
- **Precision:** Align-ETConformance gebuik makende van alignments [4]
- **Simplicity:** Inverse arc degree [8]
- **Aantal *arcs*** = A
- **Aantal *nodes*** = N = aantal places + aantal transitions van het petri net
- **Coefficient of connectivity** = A/N
- **Cyclomatric number** = A - N + 1
- **Density** = A/(N*(N-1)

Al deze evaluatiemaatstaven zijn reeds besproken in Sectie 2.3 en moeten samen een goed beeld geven over de prestaties van de verschillende trace clustering technieken en welke trace clustering technieken in combinatie met welk soort event log zijn taak goed kan afwerken.

7.4 Process discovery algoritme

Naast de selectie van de verschillende trace clustering technieken, de selectie van de event logs en de evaluatiemaatstaven heeft ook de selectie van het process discovery algoritme een effect op de uiteindelijke resultaten. In deze studie wordt gebruik gemaakt van de Inductive Miner [40]. De keuze voor de Inductive Miner is er gekomen omdat het enerzijds sound procesmodellen garandeert en makkelijk uitbreidbaar is zodat het ook nog met de inzichten in de toekomst gebruikt kan worden. De Inductive Miner heeft met andere woorden de neiging om procesmodellen te creëren met een hoge fitness en een mindere precision. Langs de andere kant is het één van de proces discovery algoritmen die geïntegreerd is in PM4PY [55]. Dat is nodig omdat de verdere analyses, na het clusterproces, gedaan worden met behulp van PM4PY [55]. Hoe het gehele analyseproces in zijn werk is gegaan wordt kort toegelicht in de volgende subsectie.

7.5 Het analyseproces

Het analyseproces bestaat uit verschillende stappen zoals eerder werd aangehaald in Figuur 13. In deze subsectie worden de verschillende stappen doorlopen. Concreet gaat het onder andere over de exploratieve data-analyse, het opstellen van verschillende configuraties van de geselecteerde trace clustering technieken en het analyseren van de clusters aan de hand van de inductive miner en de geselecteerde kwaliteitsmaatstaven. Het hele analyseproces van deze vergelijkende analyse wordt in deze subsectie besproken.

Eerst wordt er een korte exploratieve data-analyse gedaan in R aan de hand van de package BupaR [34, 59]. Enkele bevindingen van deze korte analyse zijn reeds beschreven in Subsectie 7.2. Nadat een inzicht werd verkregen in de event logs, kon gestart worden met het effectief clusteren van de verschillende event logs. Zoals eerder aangehaald werd dit gedaan in ProM 5 en Prom 6 [63]. ProM 5

kan echter enkel werken met het .mxml-bestandsformaat. Om de event logs, die standaard beschikbaar zijn in het modernere .xes-bestandsformaat, om te vormen naar .mxml-bestanden werd gebruik gemaakt van Disco [58]. Omdat er slechts beperkte computerkracht beschikbaar was, werd dit een bijkomende beperking van de studie. Door deze beperking konden niet de volledige event logs als input gebruikt worden voor de verschillende clusteralgoritmen, maar moest er gewerkt worden met samples. De specifieke gegevens van deze samples worden weergegeven in Tabel 2. Het aantal cases is bepaald op basis van de duurtijd van het SOM trace clustering algoritme waarbij gestart werd met de volledige event log en zo stelselmatig het aantal cases verminderd werd totdat de duurtijd voor dit trace clustering algoritme onder de 2 uur uitkwam.

Tabel 2: Details met betrekking tot de samples van de event logs

Event log	Aantal cases	Percentage cases
Helpdesk	500	11%
Travel Permit (BPIC 2020)	500	7%
Hospitaal (BPIC 2011)	225	19,7%

Na het bepalen van het aantal cases kon gestart worden met het clusteren. Zoals eerder aangehaald bevatten veel, zo niet alle, trace clustering algoritmen verschillende parameters die dienen ingesteld te worden door de analist. Deze parameters kunnen een groot effect hebben uit de uiteindelijke resultaten. Omdat deze studie onmogelijk alle parameters kon wijzigen en alle mogelijke combinaties kon opnemen in de studie, is er een keuze gemaakt. Eerst en vooral is er geopteerd om de standaardconfiguratie, die ingesteld is geweest door de auteurs van de plug-ins, altijd op te nemen in de studie. Daarnaast is er geprobeerd om over de clusteralgoritmen heen een lijn te trekken zodat deze zo goed als mogelijk met elkaar vergeleken kunnen worden. Omwille van deze reden is er geopteerd om elk clusteralgoritme uit te voeren met als aantal clusters 3 en 6. Een uitzondering hierop is het clusteralgoritme K-means waar het aantal clusters 3, 6 én 4 is omdat 4 clusters hier een standaardconfiguratie is. Een volledig overzicht van de verschillende configuraties kan terug gevonden worden in Tabel 3. Hier dient echter een kanttekening bij gemaakt te worden. Wegens hierboven aangehaalde beperkingen met betrekking tot onder andere de computerkracht kon het SOM trace clustering algoritme enkel met succes uitgevoerd worden voor de event log van de helpdesk. Bij de andere twee event logs nam dit algoritme steeds meer dan 2u in beslag waardoor dit algoritme buiten beschouwing werd gelaten bij de event logs travel permit en het ziekenhuis.

Tabel 3: De configuraties van de verschillende trace clustering technieken.

Trace clustering algoritme	Parameter 1	Waarden	Parameter 2	Waarden	Aantal clusters
Active Trace Clustering	Trace Selectiemethode	Frequentie	Fitness	1	3 6
				0,8	3 6
		MRA	Fitness	1	3 6
				0,8	3 6
Profiles: K-means	Afstandsmaatstaf	Euclidean			3 4 6
Sequence alignment					3
Profiles: SOM	Breedte-Hoogte	1-3			Algoritme bepaald aantal clusters
		2-3			
		4-4			
		5-5			

Na het clusteren werden de verschillende clusters telkens geëxporteerd zodat deze opnieuw gebruikt konden worden in verdere analyses. Deze analyses werden gedaan met behulp van PM4PY [55]. Hierbij werd aan de hand van de inductive miner [40] een petri net geëxtraheerd waarna de verschillende maatstaven werden berekend. Op basis van deze verschillende maatstaven werden vervolgens vergelijkende analyses in R gedaan die besproken zullen worden in de volgende sectie. Om per clusteralgoritme en per configuratie een duidelijk beeld te krijgen over de verschillende kwaliteitsmaatstaven heen wordt er voor elke configuratie en voor elke maatstaf het minimum, het gemiddelde, het maximum en het gewogen gemiddelde berekend op basis van de waarden die voortvloeien uit de verschillende clusters.

8 Resultaten

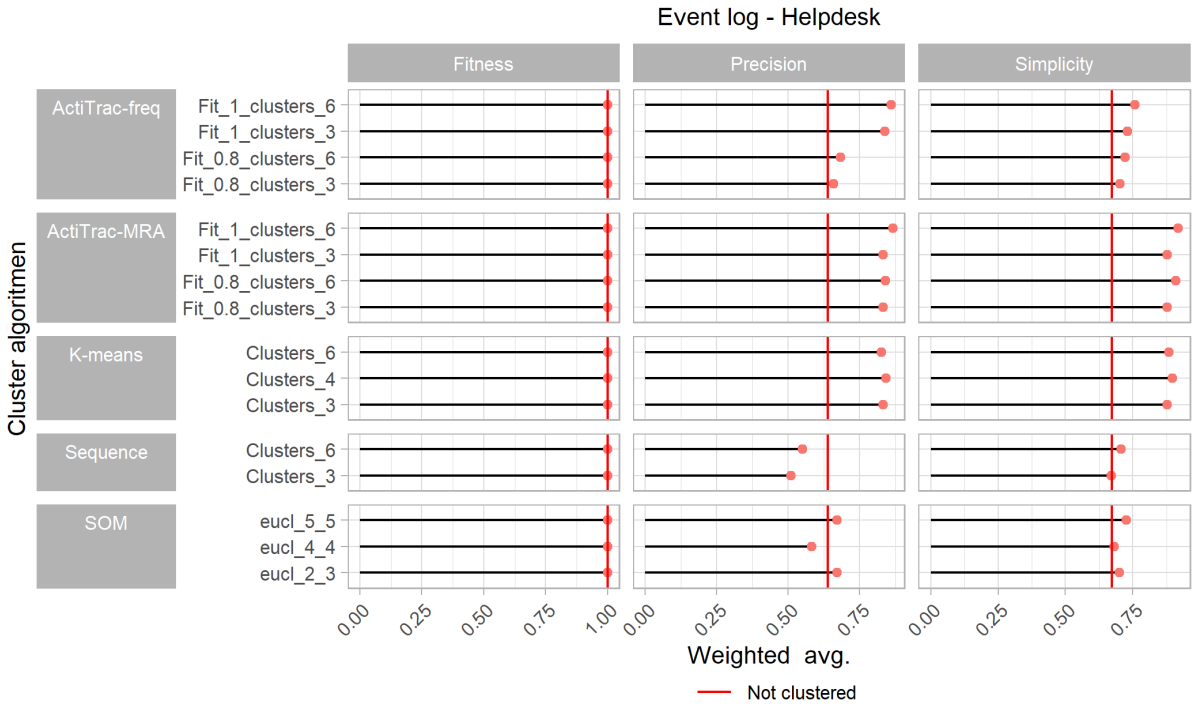
In deze sectie worden de resultaten besproken van de verschillende analyses die doorheen de studie gemaakt zijn. Er is voor geopteerd om de verschillende trace clustering technieken eerst te vergelijken aan de hand van de verschillende event logs waarbij steeds op bepaalde belangrijke aspecten zal worden ingezoomd. Na de drie verschillende event logs apart behandeld te hebben, waarbij eerst de meest eenvoudige event log besproken wordt om zo over te gaan tot de meer complexe event logs, zal eveneens een algemene vergelijking van de verschillende trace clustering technieken besproken worden. Deze algemene vergelijking kan teruggevonden worden in Sectie 9.

8.1 Event log - Helpdesk

Zoals eerder is aangehaald, is het belangrijk om bewust te zijn van het feit dat er in de vergelijkende analyse vier verschillende trace clustering technieken, elk met verscheidene configuraties, met elkaar vergeleken worden. In een eerste grafiek, zie Figuur 15, worden de verschillende clusteralgoritmen met elk hun verschillende configuraties vergeleken aan de hand van de klassieke kwaliteitsmaatstaven fitness, precision en simplicity. Op deze grafiek wordt het gewogen gemiddelde voor de verschillende maatstaven getoond en worden de niet geclusterde waarden afgebeeld als een referentielijn.

Wat misschien als eerste opvalt is het feit dat zowel de niet geclusterde event log als alle geclusterde sub-event logs een fitness van exact 1 hebben. Dit kan verklaard worden door de keuze van het process discovery algoritme dat gebruikt wordt bij deze analyse. De Inductive Miner heeft immers de neiging om modellen te extraheren met een hoge fitness en een mindere precision. Bij de volgende maatstaf, precision, is meteen een ander patroon zichtbaar. Waar de niet geclusterde event log een precision heeft van 0,64 wordt in de grafiek duidelijk dat deze maatstaf afhangt van welk clusteralgoritme bekeken wordt. De waarden van deze kwaliteitsmaatstaf liggen over alle clusteralgoritmen heen tussen 0,51 en 0,92. Een ander resultaat kan opgemerkt worden bij het clusteralgoritme Active Trace Clustering waarbij de selectiemethode gebaseerd is op de frequentie van de traces. Bij de resultaten die voortvloeien uit het daarnet genoemde clusteralgoritme kan opgemerkt worden dat de precision van de configuraties met een target fitness van 1 beduidend hoger ligt dan de precision met een target fitness van 0,8. Dit patroon kan minder duidelijk geobserveerd worden als de selectiemethode verandert wordt naar MRA. Tot slot valt bij de precision maatstaf op dat het resultaat best sterk afhangt van het clusteralgoritme. Zo daalt de precision als het Sequence clustering algoritme gebruikt wordt en in bepaalde configuraties van het SOM clusteralgoritme. Voor alle andere clusteralgoritme gaat de precision er wel op vooruit waarbij de hoogste waarden behaald worden bij het Active Trace Clustering algoritme met de selectiemethode MRA. De laatste klassieke maatstaf is simplicity. Naast het verhogen van de precision is één van de doelen van trace clustering om de procesmodellen eenvoudiger, begrijpbaarder te maken en dus met andere woorden deze maatstaf te verhogen. Alle trace clustering algoritmen lijken hierin te slagen. Echter zijn ook hier verschillen. De gewogen gemiddelden schommelen bij deze maatstaf tussen 0,67 en 0,96. Net als bij precision presteert het Sequence clusteralgoritme ook hier het minst goed. De beste leerlingen van de klas is hier het Active Trace Clustering algoritme met de selectiemethode MRA en K-Means.

Om de belangrijkste doelstelling van trace clustering, het eenvoudiger maken van de procesmodellen, nader te bekijken zijn er ook nog andere maatstaven berekend voor de verschillende configuraties. Deze worden weergegeven in Figuur 16. Al deze simplicity metrics, buiten density en simplicity zelf, dienen, om een eenvoudiger procesmodel te weerspiegelen, lager te zijn dan de niet geclusterde data. Merk op dat de simplicity maatstaf in deze grafiek dezelfde is als de simplicity maatstaf die reeds werd

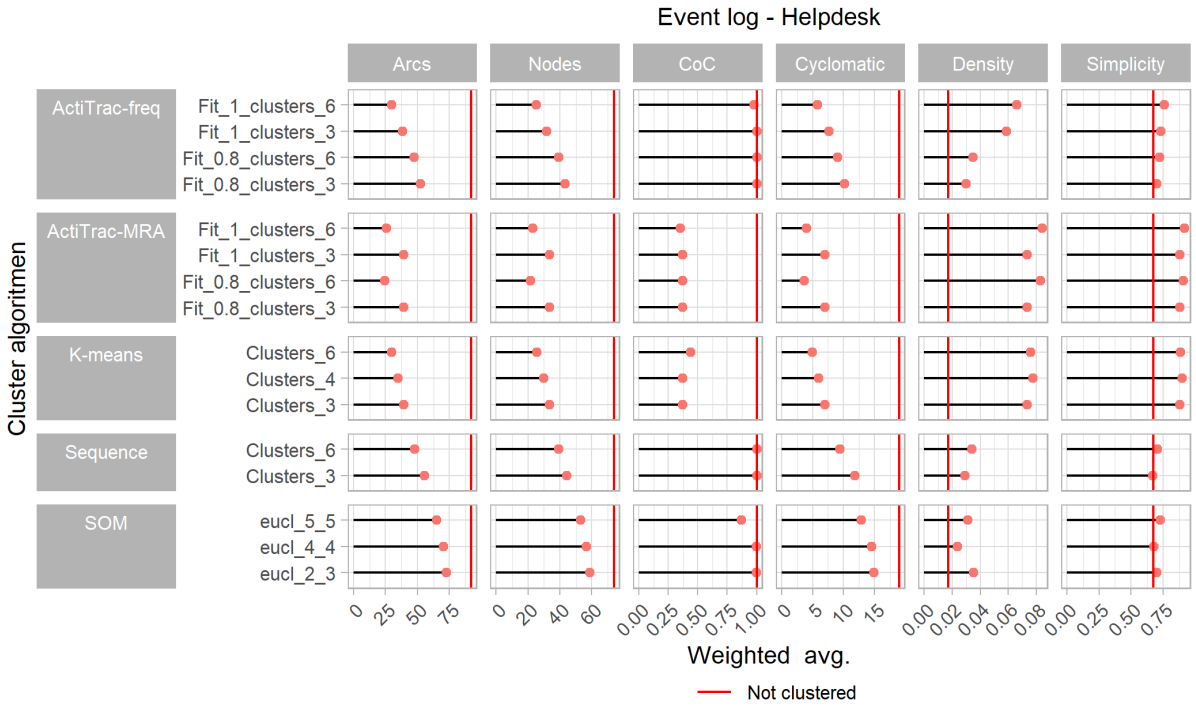


Figuur 15: De klassieke kwaliteitsmaatstaven afgetoetst tegen verschillende configuraties van trace clustering algoritmen (Event log - helpdesk).

aangehaald in Figuur 15. In Figuur 16 staat *CoC* voor *Coefficient of Connectivity* en staat *Cyclomatic* voor *Cyclomatic Number*. Algemeen kan besloten worden dat alle clusteralgoritmen, op het Sequence clusteralgoritme na, er steeds in slagen om een betere score te behalen dan de niet geclusterde data. De lagere waarden voor de eerste twee maatstaven, het aantal arcs en nodes, geeft sterk aan dat de clusteralgoritmen wel degelijk gelijkaardige traces proberen samen te zetten in één cluster zodat niet noodzakelijk alle activiteiten en alle mogelijke traces van de event log in elke cluster moeten voorkomen. Als de clusteralgoritmen geen effect zouden hebben en dus gewoon random traces zouden toewijzen, zouden meer clusters alle of zeker bijna alle activiteiten bevatten. De *Coefficient of Connectivity* is beduidend lager bij twee clusteralgoritmen. Concreet gaat het dan over Active Trace Clustering met de selectiemethode MRA en K-means clustering. Eenzelfde patroon kan teruggevonden worden voor de maatstaf *Cyclomatic Number* hoewel het bij deze maatstaf minder uitgesproken is.

Als zowel de klassieke maatstaven als de extra simplicity maatstaven samen bekeken worden valt op dat de clusteralgoritmen Active Trace Clustering met de selectiemethode MRA en K-means clustering op de meeste kwaliteitsmaatstaven het beste scoren en dus het grootste verschil maken met de niet geclusterde event log. Echter kan opgemerkt worden dat alle trace clusterign algoritmen voor verbetering zorgen.

Trace clustering technieken zorgen voor verschillende clusters en dus ook voor verschillende procesmodellen. In de grafieken die net besproken werden is gekozen om het gewogen gemiddelde te berekenen voor de verschillende maatstaven. Echter kan het ook nuttig zijn om de spreiding over de verschillende clusters heen te bekijken. In Figuur 22 en Figuur 23, die teruggevonden kunnen worden in de appendix, worden de minimum- en maximumwaarden weergegeven samen met het gemiddelde. Deze grafieken geven weer dat er over de clusters heen een relatief grote spreiding is van zowel de klassieke maatstaven als de maatstaven gelinkt aan simplicity en dat voor alle trace clustering technieken.



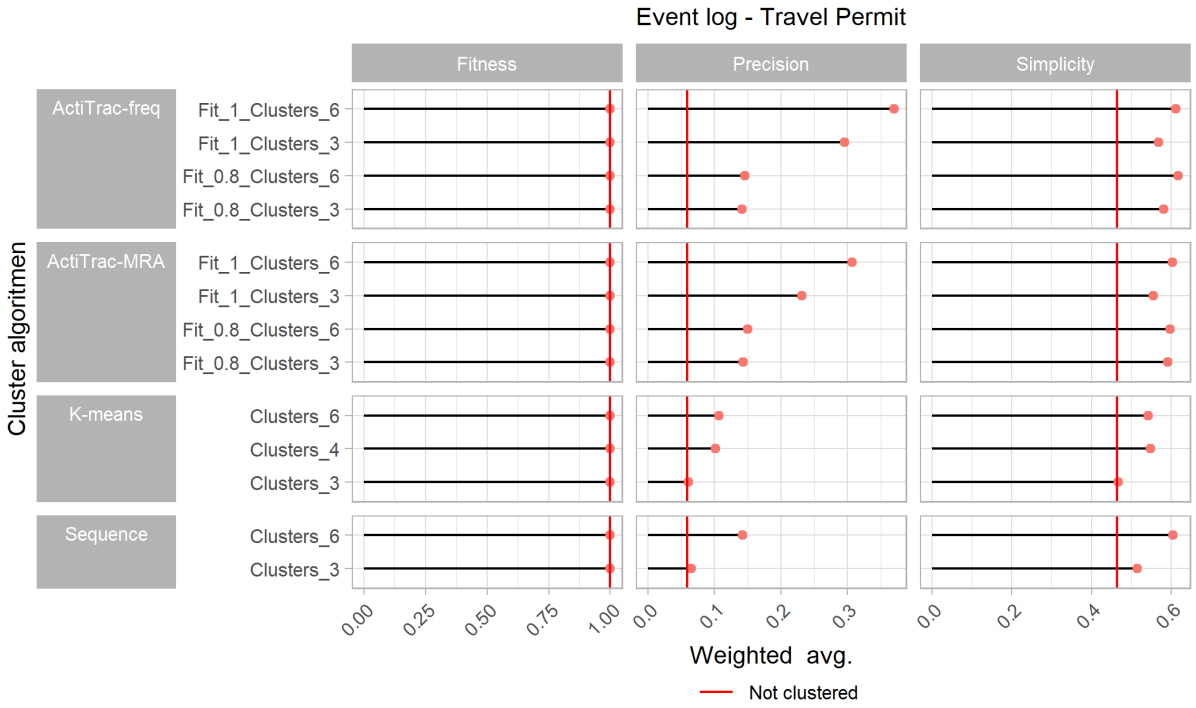
Figuur 16: Kwaliteitsmaatstaven met betrekking tot simplicity afgetoetst tegen verschillende configuraties van trace clustering algoritmen (Event log - helpdesk).

8.2 Event log - Travel permit (BPIC 2020)

In de vorige subsectie werden de resultaten van de meest eenvoudige event log van deze studie bekeken. De trace clustering technieken bleken over het algemeen geschikt om zowel de precision als de simplicity, met al haar maatstaven, te verhogen. Echter hebben trace clustering technieken een nog groter potentieel bij nog complexere processen. In deze subsectie worden daarom de resultaten van de volgende event log, Travel Permit, besproken. Dit wordt gedaan aan de hand van dezelfde grafieken.

Het is belangrijk op te merken dat de gewogen gemiddelden van zowel de precision als de simplicity van de niet geclusterde data beduidend lager zijn dan het geval was bij de log van de helpdesk. Dit fenomeen wordt visueel weergegeven in Figuur 17. Waar de gewogen gemiddelden van precision en simplicity van de event log *helpdesk* respectievelijk 0,63 en 0,67 waren, is dat voor de event log BPIC 2020 respectievelijk 0,06 en 0,46. Met andere woorden is er bij deze log veel ruimte voor verbetering. In Figuur 17 valt op dat de maatstaf fitness opnieuw voor alle configuraties van alle varianten gelijk is gebleven op de waarde 1. Voor de maatstaf precision wordt er meer variatie waargenomen. Waar de niet geclusterde data een precision heeft van 0,06 schommelt de precision van de trace clustering technieken tussen 0,07 en 0,37. Er kan gesproken worden van een duidelijke verbetering, maar een waarde van 0,37 voor precision is nog steeds relatief laag. Dit betekent dat de procesmodellen die voortvloeien uit de trace clustering technieken gedrag zullen tonen dat helemaal niet aanwezig was in de event log. Als analist van het procesmodel is het moeilijk om dergelijke resultaten betekenisvol te interpreteren. Als de analist bijvoorbeeld gedrag opvalt dat in strijd is met bepaalde compliance regels zal hij of zij alsnog moeten nagaan of dit gedrag zich al dan niet in de werkelijkheid heeft afgespeeld.

Echter is het mogelijk dat deze mindere waarden verklaard kunnen worden door een grote spreiding binnen de clusters van een bepaalde configuratie. Om dit na te gaan werd het minimum, het gemiddelde en het maximum voor elke configuratie bepaald en afgebeeld in Figuur 18. In deze grafiek zijn er



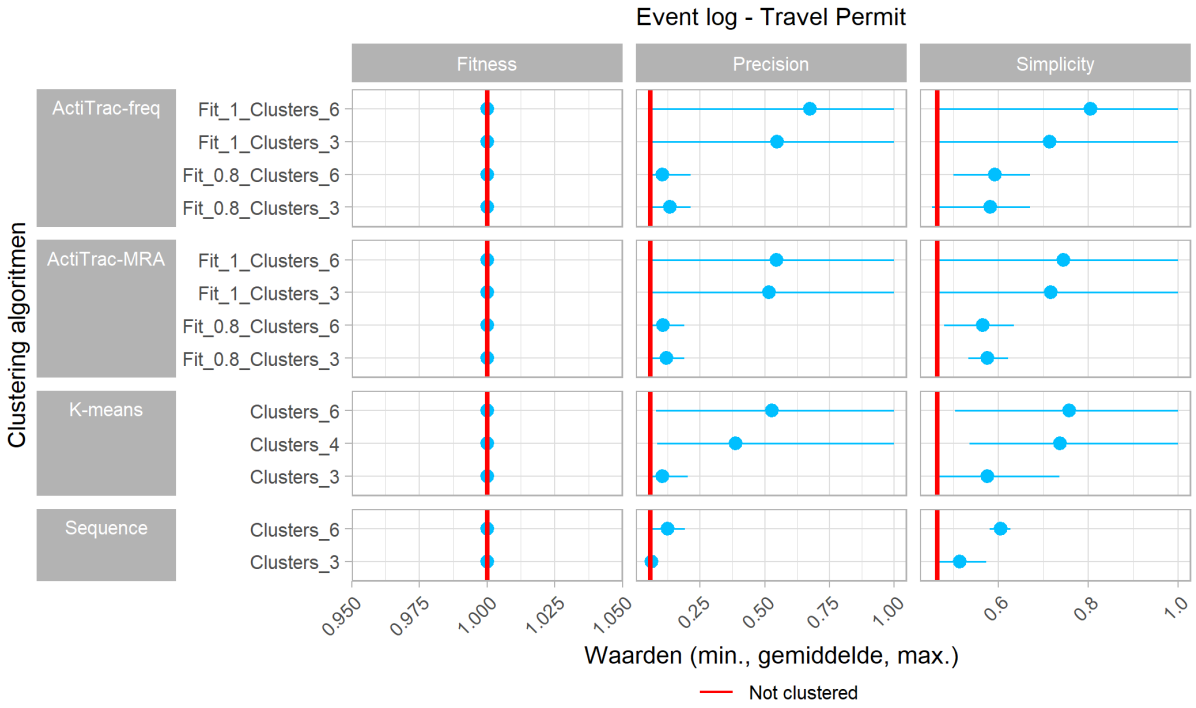
Figuur 17: Klassieke kwaliteitsmaatstaven afgetoetst tegen verschillende configuraties van trace clustering algoritmen (Event log - Travel Permit).

6 configuraties die een maximumwaarde voor precision van 1 behalen. Als er dieper ingegaan wordt op deze data blijkt dat de clusters die een precision halen van 1, slechts tussen de 0,2% en 4,4% van de cases bevatten. In absolute aantallen vertaalt zich dit in 1 case tot 22 cases. Dit verklaart ook meteen waarom het gewogen gemiddelde uit Figuur 17 relatief laag uitvalt. Voor simplicity wordt een vergelijkbaar resultaat gevonden.

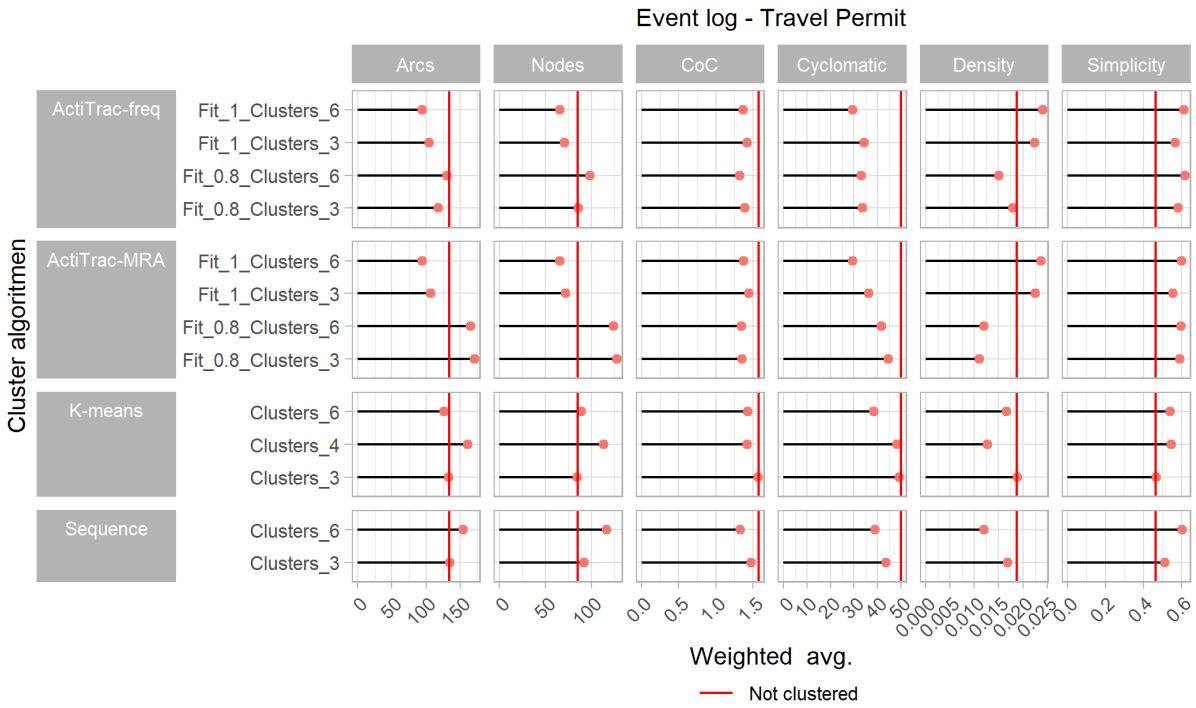
Tot slot werd er voor deze event log ook nog gekeken naar andere maatstaven die gelinkt zijn aan de complexiteit van het procesmodel. Om deze maatstaven nader te bekijken wordt de grafiek in Figuur 19 gebruikt. Voor alle maatstaven die getoond worden in de grafiek vallen wisselende resultaten op waarbij de configuratie bepaald in welke mate de trace clustering technieken verschillen van de niet geclusterde data. De trace clustering technieken slagen er bij deze event log dus niet volledig in om de procesmodellen eenvoudiger te maken door ze op te splitsen. Ook bij deze event log kan een grote spreiding tussen de clusters met betrekking tot de simplicity maatstaven vastgesteld worden. Deze grafiek kan teruggevonden worden in Figuur 24 in de appendix.

8.3 Event log - Ziekenhuis (BPIC 2011)

De laatste event log die geanalyseerd werd in de vergelijkende analyse is de event log van de BPIC 2011 over de gynaecologische afdeling in een ziekenhuis. Opnieuw worden eerst de klassieke kwaliteitsmaatstaven bekeken om een eerste beeld te krijgen van zowel de originele event log als de geclusterde sublogs. Als er in de vorige twee event logs een patroon kon gevonden worden dat de precision en simplicity dalen naarmate de event log complexer wordt, dan kan dit patroon met zekerheid doorgetrokken worden. De precision van de niet geclusterde data ligt namelijk heel dicht bij 0 zoals te zien is in Figuur 20. Meer concreet is de precision van de niet geclusterde data gelijk aan 0,003. Dit is beduidend lager dan bij de vorige twee besproken event logs. De hoogste waarde voor het ge-



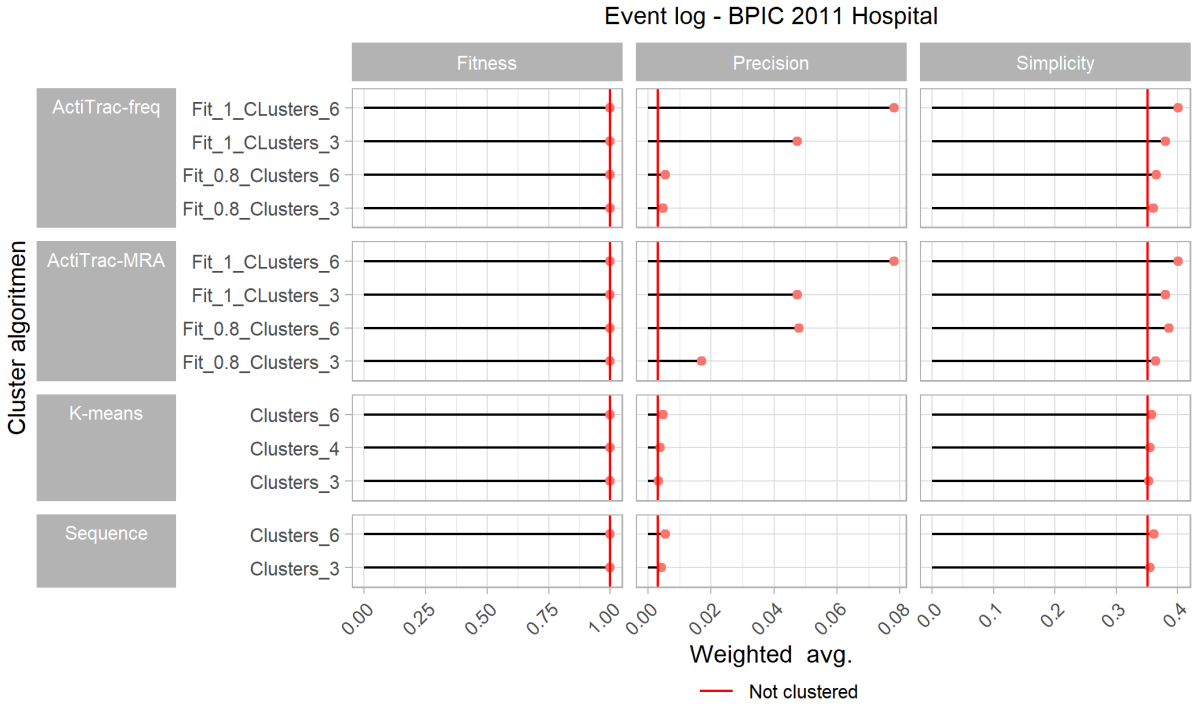
Figuur 18: De minimumwaarden, het gemiddelde en de maximumwaarden van de verschillende configuraties voor de klassieke kwaliteitsmaatstaven (Event log - Travel Permit).



Figuur 19: Kwaliteitsmaatstaven met betrekking tot simplicity afgetoetst tegen verschillende configuraties van trace clustering algoritmen (Event log - Travel Permit).

wogen gemiddelde van de verschillende trace clustering technieken wordt bereikt bij de configuratie *ActiTrac-freq Fit 1 clusters 6*. Deze configuratie bereikt een gewogen gemiddelde voor precision van 0,08. Uit de grafiek valt op te merken dat er bij het algoritme *Active Trace Clustering* merkbare ver-

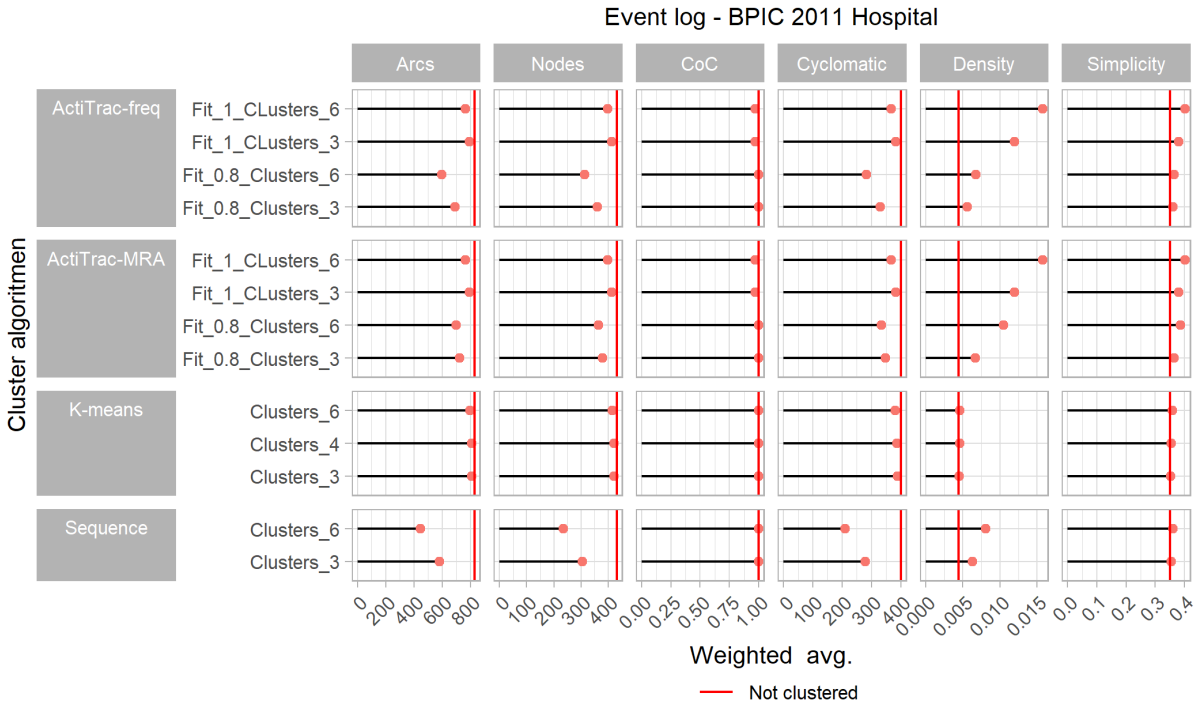
schillen zijn tussen de configuraties met 3 en met 6 clusters. Dit valt waarschijnlijk te verklaren door de grote complexiteit en variëteit in de event log. De oorspronkelijke log bevatte immers bijna evenveel traces als cases waardoor het mogelijks moeilijk is om gelijkenissen te vinden tussen dit groot aantal traces. Deze moeilijkheid komt minder tot uiting bij een groter aantal clusters. Deze moeilijkheid uit zich ook in het interval, het minimum, het gemiddelde en het maximum, van de verschillende clusters voor de verschillende maatstaven. De grafiek die deze spreiding weergeeft kan teruggevonden worden in Figuur 25.



Figuur 20: Kwaliteitsmaatstaven met betrekking tot de klassieke maatstaven afgetoetst tegen verschillende configuraties van trace clustering algoritmen (Event log - BPIC 2011 Ziekenhuis).

Naast de klassieke maatstaven zijn de simplicity maatstaven in deze vergelijkende analyse minstens zo belangrijk omwille van het oorspronkelijke doel van trace clustering. De verschillende simplicity maatstaven die geselecteerd zijn voor deze analyse worden weergegeven in Figuur 21. Bij het bekijken van deze grafiek kan opgemerkt worden dat de waarden van het gewogen gemiddelde van de verschillende trace clustering technieken bij de meeste configuraties zeer dicht liggen bij de waarde die bekomen werd bij de niet geclusterde data. De bestaande trace clustering technieken met deze configuraties zijn dus niet in staat om een zeer complexe event log, zoals deze afkomstig van de BPIC 2011, op een kwalitatieve manier op te splitsen zodat de overeenkomstige procesmodellen minder complex en dus bruikbaarder zijn dan het oorspronkelijke, niet geclusterde procesmodel. Opnieuw valt een grote spreiding tussen de clusters op zoals te zien is in Figuur 26.

Om een idee te krijgen van waarom de waarden van de gewogen gemiddelden voor de verschillende maatstaven tegenvallen is het misschien waardevol om eens te kijken naar de verdeling van de cases over de clusters heen. Bij de clusters van de configuratie met de hoogste waarden voor precision en simplicity, *ActiTrac-freq Fitness 1 Clusters 6*, is de verdeling bijvoorbeeld als volgt: 3,11%, 1,33%, 1,33%, 0,89%, 1,33% en 92,00% of in absolute aantallen 7, 3, 3, 2, 3 en 207 cases. Vervolgens kan er ook gekeken worden naar de overeenstemmende precision en simplicity. Deze zijn steeds 1 met uitzondering van de laatste, grootste cluster waar de precision en simplicity respectievelijk 0,003 en 0,351 zijn. De



Figuur 21: Kwaliteitsmaatstaven met betrekking tot simplicity afgetoetst tegen verschillende configuraties van trace clustering algoritmen (Event log - BPIC 2011 Ziekenhuis).

reden dat deze laatste cluster zo groot is, is omdat het Active Trace Clustering algoritme werkt met een zogenaamde *restcluster*. Deze *restcluster* bevat alle traces die na het clusterproces niet aan een specifieke cluster kon toegewezen worden. Echter is dit niet de volledige verklaring want ook bij andere algoritmen kan een gelijkaardig patroon van het aantal cases terug gevonden worden. De aanwezige diversiteit binnen de event log blijkt te groot te zijn wat gezien het groot aantal activiteiten ook niet verwonderlijk genoemd kan worden.

9 Discussie

In deze sectie wordt eerst met een overkoepelende blik naar de vergelijkende analyse gekeken. Welke trace clustering technieken presteren nu het beste op de verschillende maatstaven en in welke mate verschilt dit van event log tot event log. Daarnaast wordt er ook met een kritische blik naar dit werk gekeken en worden enkele beperkingen aangehaald. Tot slot worden er ook enkele aanbevelingen gedaan voor toekomstig onderzoek die kunnen voortbouwen op de resultaten die voortvloeien uit deze paper.

Om een goed beeld te krijgen van de prestaties van de verschillende trace clustering technieken is het ook belangrijk om de resultaten, die reeds per event log in de vorige sectie werden beschreven, eveneens met een overkoepelende bril te bekijken. In Tabel 4 worden de drie best presterende trace clustering technieken voor de drie verschillende event logs opgelijst. Hierbij is gekeken naar de maatstaven precision en simplicity omwille van het oorspronkelijk doel van trace clustering. Bij het nader bekijken van de tabel kunnen zeker drie zaken opvallen. Ten eerste is het opmerkelijk dat het trace cluster algoritme Active Trace Clustering van Deweerdt et al. [71] bij alle drie de event logs terugkomt met meerdere configuraties en dat zelfs 8 van de 9 opgelijste algoritmen overeenkomt met dit algoritme. Enkel de derde plaats bij de event log helpdesk wordt ingenomen door een ander trace clustering algoritme, namelijk K-means. Een tweede punt dat opvalt is dat, op twee configuraties na, alle configuraties zes clusters bevatten. Dit is niet onlogisch aangezien configuraties met meer clusters van dezelfde event log hetzelfde aantal cases en dus ook traces kunnen verdelen over meer clusters. Bijgevolg is het eenvoudiger om een meer precies en minder complex model te extraheren van de subblogs. Merk echter op dat een groter aantal clusters niet noodzakelijk makkelijker is voor de analist omdat hij of zij vervolgens zijn aandacht moet besteden aan meer procesmodellen. Tot slot kan besloten worden dat de trace clustering technieken bij de event logs helpdesk en travel permit, voor niet te verwaarlozen verbeteringen zorgen. De beste configuraties voor de event logs helpdesk en travel permit verbeteren bijvoorbeeld de simplicity met respectievelijk 0,25 en 0,15. Merk echter op dat de verbetering sterk vermindert naarmate de event logs complexer worden en eigenlijk zelfs zo goed als onbestaande is voor de meest complexe event log, hier BPIC 2011 - Hospital.

Tabel 4: Samenvattende tabel van de vergelijkende analyse.

Event log	Clusteralgoritme	Configuratie	Precision	Simplicity
Helpdesk	ActiTrac-MRA	Fitness 1 clusters 6	0,87 (+0,23)	0,92 (+0,25)
	ActiTrac-MRA	Fitness 0.8 clusters 6	0,84 (+0,20)	0,91 (+0,24)
	K-Means	Clusters 4	0,84 (+0,20)	0,90 (+0,23)
BPIC 2020 - Travel Permit	ActiTrac-Freq	Fitness 1 clusters 6	0,37 (+0,31)	0,61 (+0,15)
	ActiTrac-MRA	Fitness 1 clusters 6	0,31 (+0,25)	0,60 (+0,14)
	ActiTrac-Freq	Fitness 1 clusters 3	0,30 (+0,24)	0,57 (+0,10)
BPIC 2011 - Hospital	ActiTrac-Freq	Fitness 1 clusters 6	0,08 (+0,075)	0,40 (+0,05)
	ActiTrac-MRA	Fitness 1 clusters 6	0,08 (+0,075)	0,40 (+0,05)
	ActiTrac-MRA	Fitness 0.8 clusters 6	0,05 (+0,04)	0,39 (+0,04)

Deze masterproef had echter ook enkele beperkingen. Wegens praktische redenen, en omwille van het perspectief van een analist, werden in de vergelijkende analyse enkel trace clustering technieken in acht genomen die een publiek toegankelijke plug-in voor ProM of publiek toegankelijke R/python code ter beschikking stelden. Hierdoor is het aantal trace clustering technieken van de vergelijkende analyse beperkt. Daarnaast waren de plug-ins die beschikbaar waren in drie van de vier gevallen enkel beschikbaar in het verouderde ProM 5 framework. Ook werd er, wegens onder andere de beperkte flexibiliteit van de plug-ins, ook niet verder ingegaan op de impact van de trace representation technieken

hoewel deze een andere dimensie geven aan het meenemen van impliciete en expliciete informatie in het clusterproces. Ook was de beschikbare computerkracht, die nodig was om real-life event logs volledig te behandelen, te beperkt. In de vergelijkende analyse is er bijgevolg gebruik gemaakt van steekproeven van real-life event logs. Tot slot valt op te merken dat veel trace clustering technieken veel parameters hebben die het uiteindelijke resultaat sterk kunnen beïnvloeden. Sommige technieken vereisen zelfs specifieke detaillistische proceskennis om het de techniek te kunnen gebruiken. Dit kan voordelen hebben omdat procesexperts soms kennis bezitten die essentieel is en niet vervat zit in de data. Echter kan dit ook een zwakte zijn van deze technieken. Een procesexpert heeft namelijk onvermijdelijk een vertekend beeld van het proces. Als dit vertekend beeld meegenomen wordt in de trace clustering technieken riskeert men de clusteralgoritmen in een richting te duwen die mogelijks fout is. Het gevolg hiervan is dat ook de procesmodellen dit beeld impliciet of expliciet zullen bevatten.

Naast beperkingen zijn er ook enkele aanbevelingen voor potentieel toekomstig onderzoek voortgevloeid uit deze studie. Bij de analyse van de verschillende event logs viel duidelijk op dat de resultaten varieerden naargelang de configuratie van een specifiek trace clustering algoritme. Mogelijks is er een verband tussen de in te stellen parameters en bepaalde eigenschappen van een event log. Moest dit uit toekomstig onderzoek blijken, zou het voor de analist makkelijker zijn om aan de slag te gaan met trace clustering technieken. Als dit niet het geval zou zijn, kan men werken aan de ontwikkeling van een plug-in die binnen een interval van waarden voor de verschillende parameters zoekt naar het ideale scenario. Daarnaast is het ook noodzakelijk dat het aantal plug-ins voor trace clustering technieken uitgebreid wordt zodat de analist een grotere keuze heeft aan potentieel nog betere technieken. Tot slot is er nood aan manieren om verschillende trace representation technieken te gebruiken bij de verschillende trace clustering technieken. Het is, zoals in dit werk meermaals aangehaald, bewezen dat trace representation een impact heeft op de uiteindelijke clusterresultaten. Dit potentieel mag niet verloren gaan.

10 Conclusie

Verschillende geselecteerde trace clustering technieken zijn in deze vergelijkende studie met elkaar vergeleken aan de hand van drie verschillende event logs die elk van elkaar verschilde in complexiteit. In deze laatste sectie worden de belangrijkste bevindingen van deze studie nog eens aangehaald.

Zoals in de Introductie beschreven, en eveneens aangehaald werd bij de Probleemstelling, is het doel van trace clustering technieken voornamelijk om de precision en de simpliciteit te verhogen zodat spaghetti modellen vermeden kunnen worden. Het verhogen van de simpliciteit zou ervoor zorgen dat de geëxtraheerde procesmodellen van relatief ingewikkelde systemen opnieuw bruikbaar zouden worden voor verdere stappen in het analyseren, monitoren en verbeteren van bedrijfsprocessen. Trace clustering technieken proberen dit doel te bereiken door een complexe event log op te splitsen in verschillende sublogs zodat hierna, uit elke sublog apart, een begrijpbaar en precies model voortvloeit. Trace clustering technieken doen dit door op zoek te gaan naar traces die gelijkaardig zijn in eenzelfde cluster te plaatsen. Om de gelijkheid, *simmilarity*, van de verschillende traces in een event log te bepalen zijn er doorheen de afgelopen jaren verschillende technieken ontwikkeld. Deze werden overlopen in Sectie 5. Eveens werd duidelijk dat de literatuur de trace representation, de manier waarop een trace wordt voorgesteld en voorgeschoteld aan een clusteralgoritme, een grote impact kan hebben op het uiteindelijke clusterresultaat. Hier is vooral de vraag hoe een zo eenvoudig mogelijke representatie, in praktijk een representatie met een lage dimensionaliteit, zo veel mogelijk nuttige informatie kan bevatten.

Om te achterhalen of de verschillende trace clustering technieken wel degelijk zorgen voor procesmodellen met een hogere precision en simpliciteit, werd er in dit werk een vergelijkende analyse van verschillende technieken uitgevoerd. De eerste resultaten, van de meest eenvoudigste event log die in acht werd genomen tijdens deze studie, waren goed. De trace clustering technieken konden de event log opsplitsten in verschillende clusters zodoende dat de precision en ook de simpliciteit steeg of met andere woorden de complexiteit daalde. Echter bleken de trace clustering technieken niet volledig in staat om ook meer complexere event logs op een betekenisvolle manier te splitsen in verschillende clusters. Het verschil tussen de niet geclusterde data en het gewogen gemiddelde over de verschillende clusters heen werd kleiner, voor de event log *Travel Permit*, en zelfs zo goed als onbestaande bij de event log van het ziekenhuis. De vergelijkende analyse bracht ook naar voren dat het trace clustering algoritme Active Trace Clustering van Deweerdt et al. [71] de grootste impact heeft bij verschillende types van event logs. Bij dit clusteralgoritme was het effect op de precision en simplicity steeds het grootste.

Om trace clustering naar een volgend niveau te brengen is bijkomend onderzoek nodig. In deze specifieke wetenschappelijke discipline is er nood aan duidelijkheid omtrent de in te stellen parameters. Veel trace clustering technieken bevatten tal van zelf in te stellen parameters die een groot effect kunnen hebben op de uiteindelijke clusterresultaten. Bepaalde richtlijnen die de analist kunnen helpen bij het instellen van deze parameters zouden het werk van de analist makkelijker maken. Daarnaast is het eveneens belangrijk dat er blijvend gewerkt wordt aan het uitbreiden van het aantal plug-ins die trace clustering technieken toepassen en aan methodes om verschillende trace representation technieken mee in acht te nemen. Momenteel laten de trace clustering technieken immers veel informatie die in de event log vervat zit ongebruikt.

Referenties

[1] Wil Aalst. „Process Mining: Discovering and Improving Spaghetti and Lasagna Processes". In: *Atmospheric Environment - ATMOS ENVIRON*. Apr 2011, p. 1–7. DOI: 10.1109/CIDM.2011.6129461.

[2] Wil M. P. van der Aalst. *Process Mining: Data Science in Action*. Berlin, Heidelberg, GERMANY: Springer Berlin / Heidelberg, 2016. ISBN: 978-3-662-49851-4. URL: http://ebookcentral.proquest.com/lib/ubhasselt/detail.action?docID=4505537 (bezocht op 17-11-2021).

[3] Wil M. P. van der Aalst. „Mediating between modeled and observed behavior: The quest for the "right" process". In: *IEEE 7th International Conference on Research Challenges in Information Science (RCIS)* (mei 2013), p. 1–12. DOI: 10.1109/RCIS.2013.6577675.

[4] Arya Adriansyah e.a. „Measuring precision of modeled behavior". In: *Information systems and e-Business Management* 13.1 (feb 2015). Publisher: Springer, p. 37–67. DOI: 10.1007/s10257-014-0234-7.

[5] Adriano Augusto e.a. „Automated Discovery of Process Models from Event Logs: Review and Benchmark". In: *IEEE Transactions on Knowledge and Data Engineering* 31.4 (mei 2017), p. 686–705. DOI: 10.1109/TKDE.2018.2841877.

[6] Adriano Augusto e.a. „Split Miner: Discovering Accurate and Simple Business Process Models from Event Logs". In: *2017 IEEE International Conference on Data Mining (ICDM)*. ISSN: 2374-8486. Singapore, nov 2017, p. 1–10. DOI: 10.1109/ICDM.2017.9.

[7] Alessandro Berti en Wil MP van der Aalst. „Reviving Token-based Replay: Increasing Speed While Improving Diagnostics." In: *ATAED@ Petri Nets/ACSD*. Aug 2019, p. 87–103.

[8] Fabian Rojas Blum. „Metrics in process discovery". In: *Tech. Rep. Technical Report TR/DCC-2015-6, Computer Science Dept., University of Chile* (2015).

[9] Seppe vanden Broucke en Jochen Weerdt. „Fodina: A robust and flexible heuristic process discovery technique". In: *Decision Support Systems* 100 (apr 2017). DOI: 10.1016/j.dss.2017.04.005.

[10] Seppe vanden Broucke. „Advances in Process Mining: Artificial negative events and othertechniques". In: 2014.

[11] Hong-Nhung Bui e.a. „A Compact Trace Representation Using Deep Neural Networks for Process Mining". In: *2019 11th International Conference on Knowledge and Systems Engineering (KSE)*. ISSN: 2164-2508. Okt 2019, p. 1–5. DOI: 10.1109/KSE.2019.8919355.

[12] JCAM Buijs. „Flexible evolutionary algorithms for mining structured process models". In: *Technische Universiteit Eindhoven* 57 (okt 2014). DOI: 10.6100/IR780920.

[13] Joos CAM Buijs, Boudewijn F van Dongen en Wil MP van Der Aalst. „On the role of fitness, precision, generalization and simplicity in process discovery". In: *OTM Confederated International Conferences"On the Move to Meaningful Internet Systems"*. Springer, 2012, p. 305–322. DOI: 10.1007/978-3-642-33606-5_19.

[14] Jorge Cardoso. „How to measure the control-flow complexity of web processes and workflows". In: *Workflow handbook* 2005 (2005). Publisher: Future Strategies Inc, p. 199–212.

[15] Jorge Cardoso. „Complexity analysis of BPEL web processes". In: *Software Process: Improvement and Practice* 12.1 (2007). Publisher: Wiley Online Library, p. 35–49. DOI: 10.1002/spip.302.

[16] Jorge Cardoso e.a. „A discourse on complexity of process models". In: *International Conference on Business Process Management*. Springer, 2006, p. 117–128. DOI: 10.1007/11837862_13.

[17] *complexity noun - Definition, pictures, pronunciation and usage notes | Oxford Advanced Learner's Dictionary at OxfordLearnersDictionaries.com*. URL: https://www.oxfordlearnersdictionaries.com/definition/english/complexity?q=complexity.

[18] Pieter De Koninck, Seppe vanden Broucke en Jochen De Weerdt. „act2vec, trace2vec, log2vec, and model2vec: Representation Learning for Business Processes". en. In: *Business Process Management*. Red. door Mathias Weske e.a. Lecture Notes in Computer Science. Sydney, Australia: Springer International Publishing, sep 2018, p. 305–321. ISBN: 978-3-319-98648-7. DOI: 10.1007/978-3-319-98648-7_18.

[19] Pieter De Koninck en Jochen De Weerdt. „Multi-objective Trace Clustering: Finding More Balanced Solutions". en. In: *Business Process Management Workshops*. Red. door Marlon Dumas en Marcelo Fantinato. Cham: Springer International Publishing, 2017, p. 49–60. ISBN: 978-3-319-58457-7. DOI: 10.1007/978-3-319-58457-7_4.

[20] Pieter De Koninck e.a. „Expert-driven trace clustering with instance-level constraints". en. In: *Knowledge and Information Systems* 63.5 (mei 2021), p. 1197–1220. ISSN: 0219-3116. DOI: 10.1007/s10115-021-01548-6. URL: https://doi.org/10.1007/s10115-021-01548-6 (bezocht op 17-11-2021).

[21] Pavlos Delias e.a. „A non-Compensatory Approach for Trace Clustering". In: *International Transactions in Operational Research* 26.5 (feb 2017), p. 1828–1846. DOI: 10.1111/itor.12395.

[22] Pavlos Delias e.a. „Improving the non-compensatory trace clustering decision process". In: *International Transactions in Operational Research* (sep 2021). DOI: 10.1111/itor.13062.

[23] Boudewijn van Dongen. „BPI Challenge 2020". In: (mrt 2020). DOI: 10.4121/uuid:52fb97d4-4588-43c9-9d04-3604d4613b51. URL: https://data.4tu.nl/collections/BPI_Challenge_2020/5065541/1.

[24] Boudewijn van Dongen. „Real-life event logs - Hospital log". In: (mrt 2011). DOI: 10.4121/uuid:d9769f3d-0ab0-4fb8-803b-0d1120ffcf54. URL: https://data.4tu.nl/articles/dataset/Real-life_event_logs_-_Hospital_log/12716513.

[25] Bruce Edmonds. „What is Complexity?-The philosophy of complexity per se with application to some examples in evolution". In: *The evolution of complexity*. Kluwer, Dordrecht, 1995.

[26] *Event Data - Process Mining*. en. URL: http://www.processmining.org/event-data.html (bezocht op 27-02-2022).

[27] Joerg Evermann, Tom Thaler en Peter Fettke. „Clustering Traces Using Sequence Alignment". en. In: *Business Process Management Workshops*. Red. door Manfred Reichert en Hajo A. Reijers. Cham: Springer International Publishing, 2015, p. 179–190. ISBN: 978-3-319-42887-1. DOI: 10.1007/978-3-319-42887-1_15.

[28] Quang-Thuy Ha, Hong-Nhung Bui en Tri-Thanh Nguyen. „A Trace Clustering Solution Based on Using the Distance Graph Model". en. In: *Computational Collective Intelligence*. Red. door Ngoc-Thanh Nguyen e.a. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2016, p. 313–322. ISBN: 978-3-319-45243-2. DOI: 10.1007/978-3-319-45243-2_29.

[29] Michael Hammer en James Champy. *Reengineering the Corporation: Manifesto for Business Revolution, A*. Zondervan, 2009.

[30] Bart Hompes e.a. „Discovering Deviating Cases and Process Variants Using Trace Clustering". In: Hasselt, Belgium, nov 2015.

[31] „IEEE Standard Glossary of Software Engineering Terminology". In: *IEEE Std 610.12-1990* (1990), p. 1–84. DOI: 10.1109/IEEESTD.1990.101064.

[32] R.P. Jagadeesh Chandra Bose. „Process mining in the large : preprocessing, discovery, and diagnostics". ISBN: 9789038631189. Phd Thesis 1 (Research TU/e / Graduation TU/e). Eindhoven: Technische Universiteit Eindhoven, 2012. DOI: 10.6100/IR730954.

[33] Anil K Jain en Richard C Dubes. *Algorithms for clustering data*. Prentice-Hall, Inc., 1988.

[34] Gert Janssenswillen. *bupaR: Business Process Analysis in R*. 2020. URL: https://CRAN.R-project.org/package=bupaR.

[35] Gert Janssenswillen. *Unearthing the Real Process Behind the Event Data: The Case for Increased Process Realism*. English. Cham: Springer International Publishing AG, 2021. ISBN: 3030707326;9783030707323;

[36] Sawsan Kanj, Thomas Brüls en Stéphane Gazut. „Shared nearest neighbor clustering in a locality sensitive hashing framework". In: *Journal of Computational Biology* 25.2 (2018). Publisher: Mary Ann Liebert, Inc. 140 Huguenot Street, 3rd Floor New Rochelle, NY 10801 USA, p. 236–250.

[37] R Kurdahl. *2.5 Performing a Building Blocks Search - Rune Kurdahl - Performing your searches*. en. URL: https://www.coursera.org/lecture/academicinfoseek/2-5-performing-a-building-blocks-search-rune-kurdahl-QjkNX (bezocht op 01-11-2019).

[38] Antti M Latva-Koivisto. „Finding a complexity measure for business process models". In: (2001). Publisher: Citeseer.

[39] Quoc Le en Tomas Mikolov. „Distributed Representations of Sentences and Documents". en. In: *Proceedings of the 31st International Conference on Machine Learning*. ISSN: 1938-7228. PMLR, jun 2014, p. 1188–1196. URL: https://proceedings.mlr.press/v32/le14.html (bezocht op 16-02-2022).

[40] Sander J. J. Leemans, Dirk Fahland en Wil M. P. van der Aalst. „Discovering Block-Structured Process Models from Event Logs - A Constructive Approach". en. In: *Application and Theory of Petri Nets and Concurrency*. Red. door José-Manuel Colom en Jörg Desel. Lecture Notes in Computer Science. Milan, Italy: Springer, jun 2013, p. 311–329. ISBN: 978-3-642-38697-8. DOI: 10.1007/978-3-642-38697-8_17.

[41] Sander J. J. Leemans, Dirk Fahland en Wil M. P. van der Aalst. „Discovering Block-Structured Process Models from Event Logs Containing Infrequent Behaviour". en. In: *Business Process Management Workshops*. Red. door Niels Lohmann, Minseok Song en Petia Wohed. Lecture Notes in Business Information Processing. Bejing, China: Springer International Publishing, aug 2013, p. 66–78. ISBN: 978-3-319-06257-0. DOI: 10.1007/978-3-319-06257-0_6.

[42] Sander JJ Leemans, Dirk Fahland en Wil MP van der Aalst. „Discovering block-structured process models from incomplete event logs". In: *International conference on applications and theory of petri nets and concurrency*. Springer, 2014, p. 91–110.

[43] Henry Lizano-Mora, Pedro R Palos-Sánchez en Mariano Aguayo-Camacho. „The evolution of business process management: A bibliometric analysis". In: *IEEE Access* 9 (2021). Publisher: IEEE, p. 51088–51105.

[44] James Manyika e.a. *Big data: The next frontier for innovation, competition, and productivity*. McKinsey Global Institute, 2011.

[45] Thomas J McCabe. „A complexity measure". In: *IEEE Transactions on software Engineering* 4 (1976). Publisher: IEEE, p. 308–320.

[46] Alex Meincheim e.a. „Combining Process Mining with Trace Clustering: Manufacturing Shop Floor Process - An Applied Case". In: *2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI)*. ISSN: 2375-0197. Nov 2017, p. 498–505. DOI: 10.1109/ICTAI.2017.00082.

[47] J Mendling. „Detection and prediction of errors in EPC business process models (Doctoral dissertation)". In: *WU Vienna University of Economics and Business Administration* (2007).

[48] Jan Mendling, Gustaf Neumann en Wil van der Aalst. „Understanding the occurrence of errors in process models based on metrics". In: *OTM Confederated International Conferences"On the Move to Meaningful Internet Systems"*. Springer, 2007, p. 113–130.

[49] Tomas Mikolov e.a. „Efficient Estimation of Word Representations in Vector Space". In: (sep 2013). arXiv: 1301.3781. URL: http://arxiv.org/abs/1301.3781 (bezocht op 16-02-2022).

[50] Jorge Munoz-Gama e.a. *Conformance checking and diagnosis in process mining*. Springer, 2016.

[51] Vahideh Naderifar, Shahnorbanun Sahran en Zarina Shukur. „A review on conformance checking technique for the evaluation of process mining algorithms". In: *TEM Journal* 8.4 (2019). Publisher: UIKTEN-Association for Information Communication Technology Education and ..., p. 1232.

[52] Hoang Nguyen e.a. „Stage-based discovery of business process models from event logs". In: *Information Systems* 84 (2019). Publisher: Elsevier, p. 214–237.

[53] Guido Ongena en Pascal Ravesteyn. „Business process management maturity and performance: A multi group analysis of sectors and organization sizes". In: *Business Process Management Journal* (2019). Publisher: Emerald Publishing Limited.

[54] K. Padron. *LibGuides: Guide to Science Information Resources: Backward & Forward Reference Searching*. en. Library Catalog: libguides.fau.edu. URL: https://libguides.fau.edu/science_resources/reference_searching (bezocht op 30-04-2020).

[55] *PM4Py - Process Mining for Python*. URL: https://pm4py.fit.fraunhofer.de/ (bezocht op 22-05-2022).

[56] Gregor Polančič en Blaž Cegnar. „Complexity metrics for process models – A systematic literature review". In: *Computer Standards & Interfaces* 51 (2017), p. 104–117. ISSN: 0920-5489. DOI: https://doi.org/10.1016/j.csi.2016.12.003. URL: https://www.sciencedirect.com/science/article/pii/S0920548916302276.

[57] Mirko Polato. „Dataset belonging to the help desk log of an Italian Company". In: (jul 2017). DOI: 10.4121/uuid:0c60edf1-6f83-4e75-9367-4c63b3e9d5bb. URL: https://data.4tu.nl/articles/dataset/Dataset_belonging_to_the_help_desk_log_of_an_Italian_Company/12675977.

[58] *Process Mining and Automated Process Discovery Software for Professionals - Fluxicon Disco*. URL: https://fluxicon.com/disco/ (bezocht op 27-05-2022).

[59] R Core Team. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing, 2021. URL: https://www.R-project.org/.

[60] Jagadeesh Chandra Bose R.P. en Wil Aalst. „Context Aware Trace Clustering: Towards Improving Process Mining Results". In: Journal Abbreviation: SDM Publication Title: SDM. Nevada, apr 2009. DOI: 10.1137/1.9781611972795.35.

[61] Laura Sánchez-González e.a. „Prediction of business process model quality based on structural metrics". In: *International Conference on Conceptual Modeling*. Springer, 2010, p. 458–463.

[62] Minseok Song, Christian Günther en Wil Aalst. *Trace Clustering in Process Mining*. Deel 17. Journal Abbreviation: Lecture Notes in Business Information Processing Pages: 120 Publication Title: Lecture Notes in Business Information Processing. Sep 2008. ISBN: 978-3-642-00327-1. DOI: 10.1007/978-3-642-00328-8_11.

[63] *start | ProM Tools*. URL: https://www.promtools.org/doku.php (bezocht op 09-04-2022).

[64] Yaguang Sun, B. Bauer en M. Weidlich. „Compound Trace Clustering to Generate Accurate and Simple Sub-Process Models". In: *Service-Oriented Computing*. Malaga: Springer, nov 2017, p. 175–190. DOI: 10.1007/978-3-319-69035-3_12.

[65] Yaguang Sun en Bernhard Bauer. „A novel heuristic method for improving the fitness of mined business process models". In: *International Conference on Service-Oriented Computing*. Springer, 2016, p. 537–546.

[66] Dalia Suša Vugec, Lucija Ivančić en Ljubica Milanović Glavan. „Business process management and corporate performance management: Does their alignment impact organizational performance". In: *Interdisciplinary Description of Complex Systems: INDECS* 17.2-B (2019). Publisher: Hrvatsko interdisciplinarno društvo, p. 368–384.

[67] Tom Thaler e.a. „A Comparative Analysis of Process Instance Cluster Techniques". en. In: *Wirtschaftsinformatik Proceedings 2015*. Osnabrück, apr 2015, p. 16.

[68] Alaa M Ubaid en Fikri T Dweiri. „Business process management (BPM): terminologies and methodologies unified". In: *International Journal of System Assurance Engineering and Management* 11.6 (2020). Publisher: Springer, p. 1046–1064.

[69] H. M. W. Verbeek e.a. „XES, XESame, and ProM 6". English. In: *Information Systems Evolution (CAiSE Forum 2010, Hammamet, Tunisia, June 7-9, 2010, Selected Extended Papers)*. Red. door P. Soffer en E. Proper. Lecture Notes in Business Information Processing. Germany: Springer, 2011, p. 60–75. ISBN: 978-3-642-17721-7. DOI: 10.1007/978-3-642-17722-4_5.

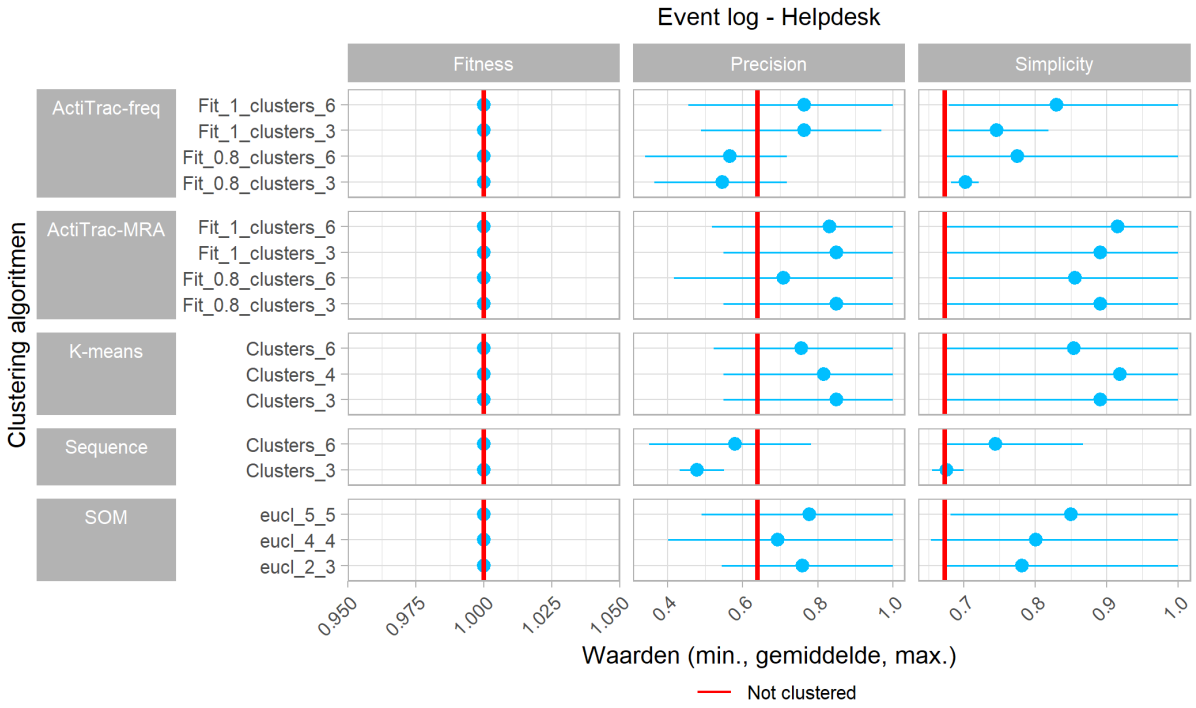
[70] Kiri Wagstaff e.a. „Constrained K-menas Clustering with Background Knowledge". In: *Proceedings of the Eighteenth International Conference on Machine Learning*. 2001, p. 577–584.

[71] Jochen Weerdt e.a. „Active Trace Clustering for Improved Process Discovery". In: *IEEE Trans. Knowl. Data Eng.* 25 (dec 2013), p. 2708–2720. DOI: 10.1109/TKDE.2013.64.

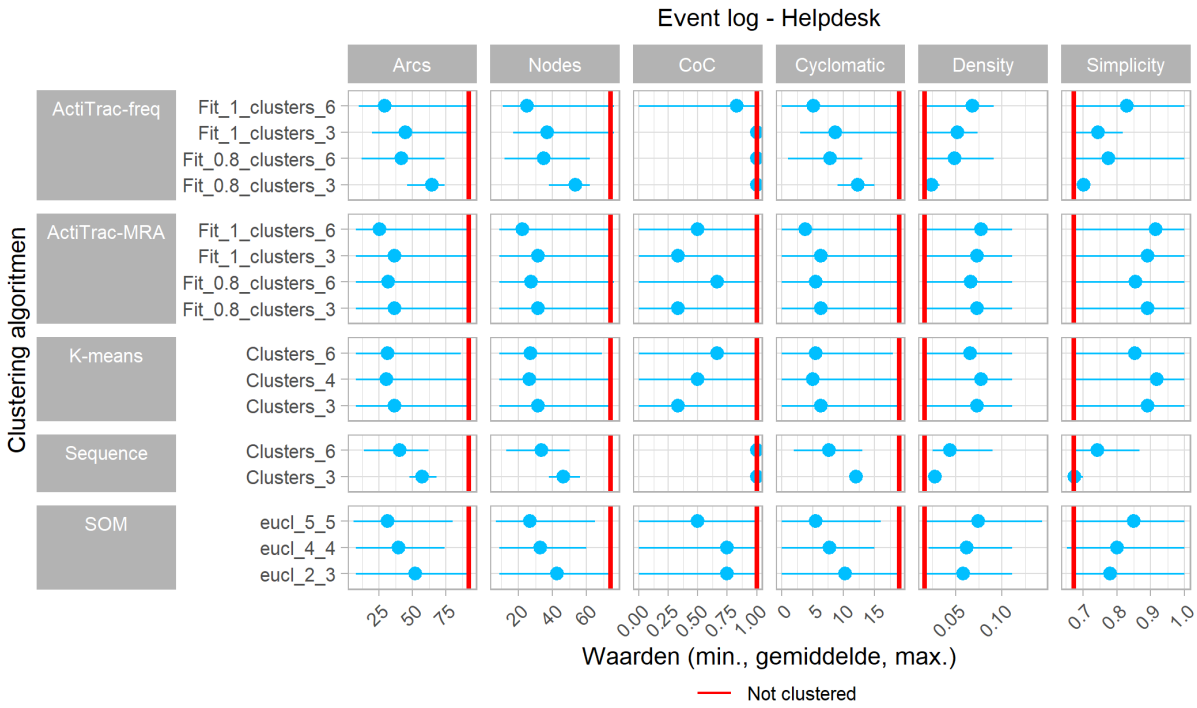
[72] Fitri Almira Yasmin, Faiza Allah Bukhsh en Patricio De Alencar Silva. „Process enhancement in process mining: A literature review". In: *CEUR workshop proceedings*. Deel 2270. Rheinisch Westfälische Technische Hochschule, 2018, p. 65–72.

[73] Fareed Zandkarimi e.a. „A Generic Framework for Trace Clustering in Process Mining". In: okt 2020. DOI: 10.1109/ICPM49681.2020.00034.

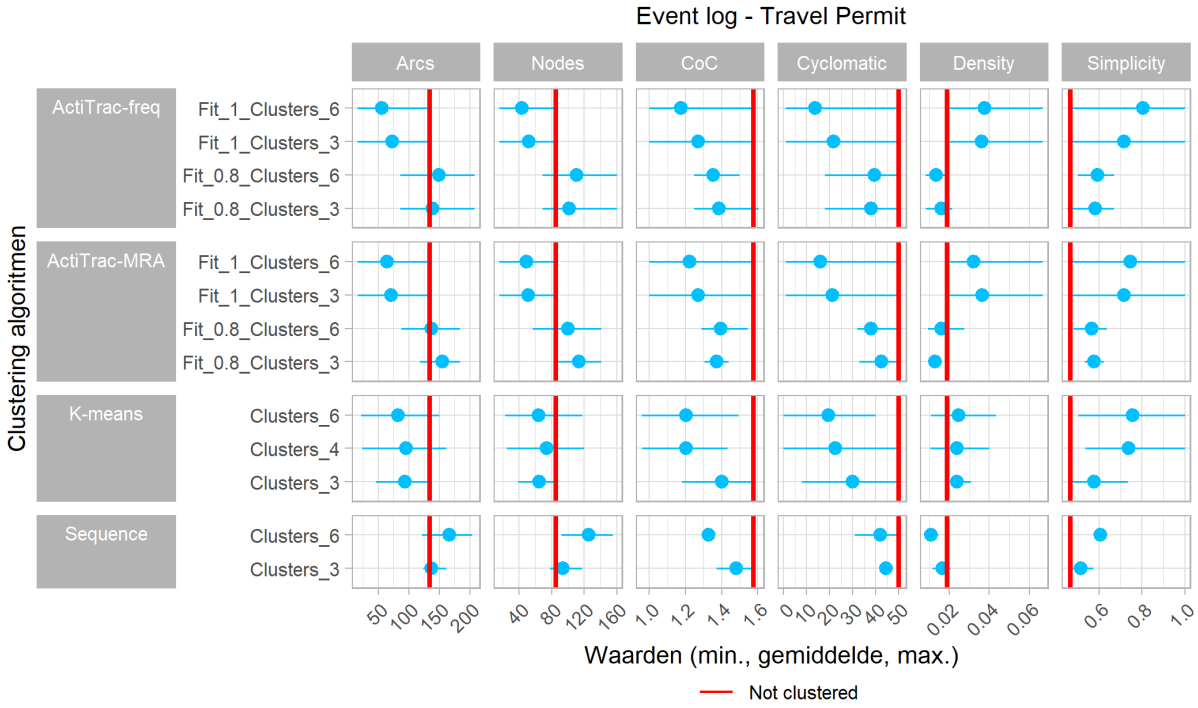
11 Appendix



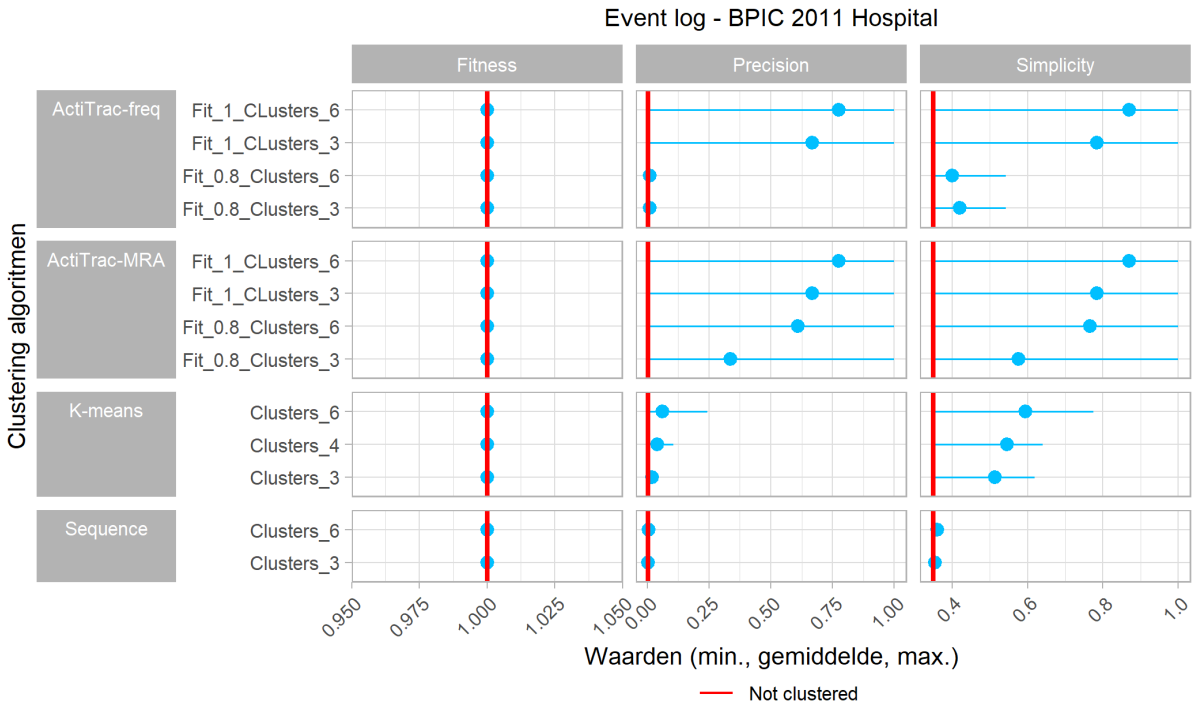
Figuur 22: De minimumwaarden, het gemiddelde en de maximumwaarden van de verschillende configuraties voor de klassieke kwaliteitsmaatstaven (Event log - helpdesk).



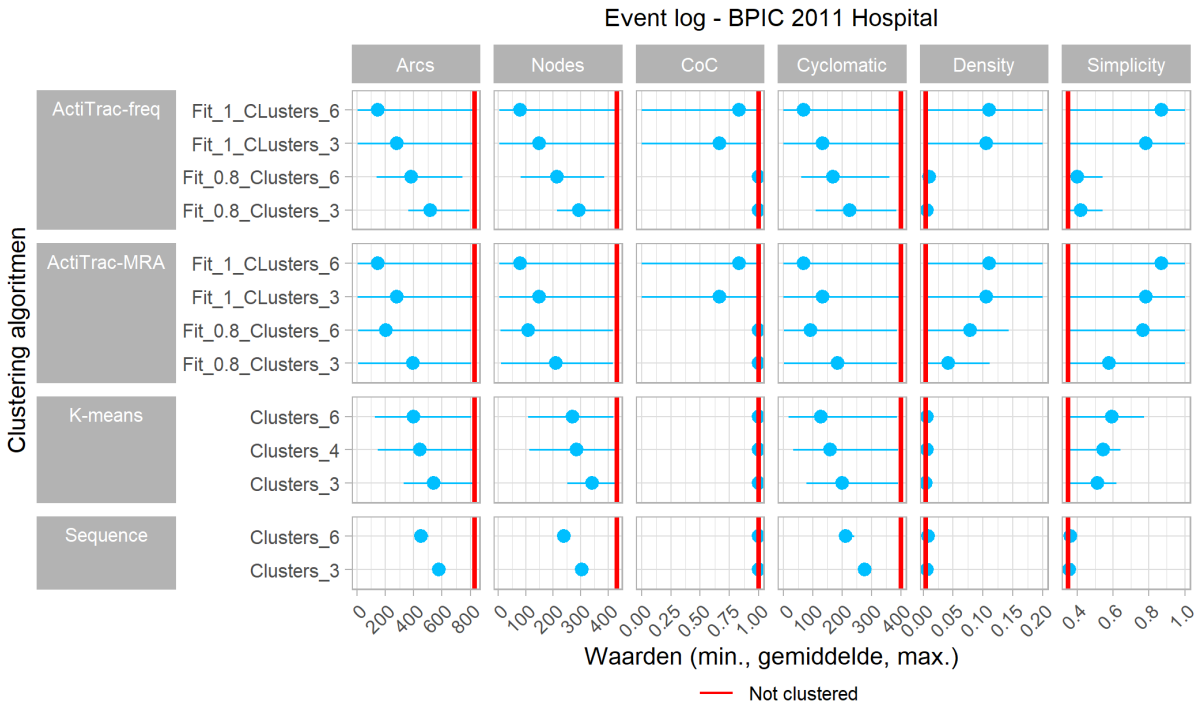
Figuur 23: De minimumwaarden, het gemiddelde en de maximumwaarden van de verschillende configuraties voor de simplicity kwaliteitsmaatstaven (Event log - helpdesk).



Figuur 24: De minimumwaarden, het gemiddelde en de maximumwaarden van de verschillende configuraties voor de simplicity kwaliteitsmaatstaven (Event log - Travel Permit).



Figuur 25: De minimumwaarden, het gemiddelde en de maximumwaarden van de verschillende configuraties voor de klassieke kwaliteitsmaatstaven (Event log - BPIC 2011 Hospital).



Figuur 26: De minimumwaarden, het gemiddelde en de maximumwaarden van de verschillende configuraties voor de simplicity kwaliteitsmaatstaven (Event log - BPIC 2011 Hospital).