TESE DE DOUTORAMENTO

BEHAVIORAL-BASED ALGORITHMS FOR PROCESS MODEL SIMPLIFICATION

Presentada por:

David Chapela de la Campa

Dirixida por:

Manuel Mucientes Molina
Manuel Lama Penín

ESCOLA DE DOUTORAMENTO INTERNACIONAL DA UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

PROGRAMA DE DOUTORAMENTO EN INVESTIGACIÓN EN TECNOLOXÍAS DA INFORMACIÓN

SANTIAGO DE COMPOSTELA
2021

DECLARACIÓN DO AUTOR DA TESE
BEHAVIORAL-BASED ALGORITHMS FOR PROCESS MODEL SIMPLIFICATION

Don David Chapela de la Campa

Presento a miña tese, seguindo o procedemento adecuado ao Regulamento, e declaro que:

1. *A tese abarca os resultados da elaboración do meu traballo.*
2. *De ser o caso, na tese faise referencia ás colaboracións que tivo este traballo.*
3. *Confirmo que a tese non incorre en ningún tipo de plaxio doutros autores nin de traballos presentados por min para a obtención doutros títulos.*
4. *A tese é a versión definitiva presentada para a súa defensa e coincide a versión impresa coa presentada en formato electrónico.*

E comprométome a presentar o Compromiso Documental de Supervisión no caso de que o orixinal non estea na Escola.

En Santiago de Compostela, 8 de xullo de 2021

Asdo. David Chapela de la Campa

AUTORIZACIÓN DO DIRECTOR/TITOR DA TESE
BEHAVIORAL-BASED ALGORITHMS FOR PROCESS MODEL SIMPLIFICATION

Don Manuel Mucientes Molina, Profesor Contratado Doutor da Área de Ciencia da Computación e Intelixencia Artificial da Universidade de Santiago de Compostela

Don Manuel Lama Penín, Profesor Titular da Área de Ciencia da Computación e Intelixencia Artificial da Universidade de Santiago de Compostela

INFORMAN:

*Que a presente tese correspóndese co traballo realizado por **Don David Chapela de la Campa**, baixo a nosa dirección/titorización, e autorizamos a súa presentación, considerando que reúne os requisitos esixidos no Regulamento de Estudos de Doutoramento da USC, e que como directores/titores desta non incorre nas causas de abstención establecidas na Lei 40/2015.*

De acordo co indicado no Regulamento de Estudos de Doutoramento, declaramos tamén que a presente tese de doutoramento é idónea para ser defendida en base á modalidade de COMPENDIO DE PUBLICACIÓNS, nos que a participación do/a doutorando/a foi decisiva para a súa elaboración e as publicacións se axustan ao Plan de Investigación.

En Santiago de Compostela, 8 de xullo de 2021

Asdo. Manuel Mucientes Molina	Asdo. Manuel Lama Penín
Director/a tese	Director/a tese

And I knew exactly what to do. But in a much more real sense, I had no idea what to do.

— Michael Scott

"Acknowledgement is the only way to keep love alive."

— Barry Long

Agradecementos

En primeiro lugar, quero expresar o meu maior agradecemento aos meus directores de tese, Manuel Mucientes Molina e Manuel Lama Penín, pola confianza depositada en min, e pola axuda brindada durante todos estes anos.

Quixera agradecer tamén ao Departamento de Electrónica e Computación, así como ao Centro de Investigación en Tecnoloxías Intelixentes (CiTIUS) da Universidade de Santiago de Compostela, pola contorna de traballo e recursos que me permitiron levar a cabo esta tese doutoral. Gracias ao Prof. Paulo Félix, director adxunto do CiTIUS durante gran parte do desenvolvemento desta tese, polo seu apoio, proximidade, e esforzo por facer do centro un lugar cómodo e agradable no que investigar. Grazas ao persoal de apoio do CiTIUS pola súa dispoñibilidade e axuda, facilitando a realización do meu traballo no centro. Aproveito tamén para agradecer ao Prof. Marlon Dumas, da Universidade de Tartu, por acollerme durante a miña estadía, así coma a todas as persoas que coñecín en Estonia e que fixeron desa etapa unha experiencia excepcional.

Gustaríame agradecer tamén a todas as entidades que financiaron o desenvolvemento desta tese doutoral, permitíndome realizala cun contrato de traballo, como se deberían realizar todas as actividades de investigación. En primeiro lugar ao programa de Formación de Profesorado Universitario do Ministerio de Educación, Cultura y Deporte (FPU16/04428 e EST19/00135), e ao Ministerio de Economía, Industria y Competitividad (TIN2017-84796-C2-1-R). Tamén á Consellería de Cultura, Educación e Ordenación Universitaria de Galicia (ED431G/08), co-financiadas polo Fondo Europeo de Desenvolvemento Rexional do programa ERDF/FEDER.

Non podería esquecerme dos compañeiros e compañeiras do laboratorio LS02, tanto aos que me acolleron nese lugar tan máxico, coma os que foron chegando e adaptándose ao xenial ambiente que tiñamos alí (rest in peace), así coma ao resto de persoas que fixeron do *Café* algo tan desexado. Agradecer especialmente a Alvarito, Víctor e Andrea, por acompañarme durante toda esta viaxe, compartindo os bos momentos, e facendo que os malos non pesasen tanto. Agradecer tamén ás compas da Asemblea de Investigadoras de Compostela, por axudarme a transformar o meu descontento coa precariedade nas primeiras etapas da investigación nunha loita para acabar con ela.

Por último, e máis importante, agradecer á miña familia polo apoio durante todos estes anos, e a todas esas persoas que formaron e forman parte da miña vida, sen as cales non sería quen son.

8 de xullo de 2021

Resumo

Os procesos son omnipresentes. Toda organización —xa sexa unha organización sen ánimo de lucro, un organismo gobernamental ou unha empresa— ten que xestionar procesos. Para mellorar estes procesos aumentando a súa eficiencia e eficacia, a súa análise e estudo do comportamento convertéronse nunha necesidade para calquera organización. Como resultado, a cantidade de datos rexistrados relacionados coa execución destes procesos viuse incrementada enormemente durante os últimos anos. Este aumento creou a necesidade de desenvolver técnicas e métodos para automatizar e mellorar a análise dos datos rexistrados.

Neste contexto, a Xestión de Procesos de Negocio (*Business Process Management* ou BPM en inglés) aparece como unha disciplina que ofrece un conxunto de métodos, técnicas e ferramentas para identificar, descubrir, analizar, redeseñar, executar e supervisar os procesos para optimizar o seu rendemento. Coa fin de mellorar continuamente o proceso, a BPM define un procedemento cíclico formado por un conxunto de etapas ou fases para guiar a análise do proceso, alimentando cada fase coa información obtida nas anteriores. O ciclo de vida da BPM comeza coa identificación do proceso, da súa arquitectura e das medidas de rendemento (*identificación do proceso*). A continuación, documéntase o estado actual do proceso (*descubrimento do proceso*), para analizalo na seguinte etapa e detectar os problemas que afectan ó rendemento do proceso (*análise do proceso*). A seguinte fase consiste en identificar os cambios para redeseñar o proceso e abordar os problemas detectados na etapa anterior (*redeseño do proceso*). Estes cambios aplícanse no proceso como parte da seguinte fase (*implementación do proceso*). Finalmente, unha vez aplicados os cambios, a última etapa consiste en supervisar o rendemento do proceso (*supervisión do proceso*) e volver iniciar o ciclo para melloralo de forma continua.

Ata hai pouco, a maioría destas etapas tiñan que facerse manualmente, o que requiría unha gran cantidade de interacción humana para deseñar o modelo que describe o comportamento

do proceso, ou para detectar os problemas que hai que mellorar. A minería de procesos (*Process Mining* ou PM en inglés) xurdiu como unha disciplina de investigación para reducir a interacción humana necesaria nestas etapas aumentando a súa automatización. A minería de procesos sitúase entre entre a aprendizaxe automática e a minería de datos por unha banda, e o modelado e análise de procesos pola outra, ofrecendo técnicas para descubrir, supervisar e mellorar os procesos empresariais extraendo coñecementos dos rexistros de eventos facilmente dispoñibles nos sistemas actuais. Desta forma, a minería de procesos establece un vínculo entre os datos rexistrados e os modelos de proceso que describen o comportamento do proceso.

Para establecer este vínculo, as técnicas pertencentes á minería de procesos adóitanse clasificar en tres tipos principais. O primeiro tipo é o *Descubrimento*. As técnicas de descubrimento toman como entrada un rexistro de eventos coa información rexistrada sobre o proceso, e producen un modelo de proceso que describe —completa ou parcialmente— o comportamento observado. Aínda que a maioría das técnicas de descubrimento non teñen en conta ningunha información a priori, algunhas técnicas requiren este coñecemento para producir o modelo de proceso. O segundo tipo é a *Análise de Conformidade*. As técnicas de comprobación da conformidade comparan un modelo de proceso existente —xa sexa descuberto ou deseñado manualmente— cun rexistro de eventos do mesmo proceso. Estas técnicas utilízanse para detectar desviacións entre o comportamento (real) rexistrado no rexistro e o comportamento soportado polo modelo. Estas desviacións poden indicar tanto problemas no modelo de proceso que deben ser corrixidos, como incidentes que xa ocorreron e que deben ser evitados no futuro. Por exemplo, tomemos un proceso de fabricación no que a calidade dun produto debe comprobarse nun punto concreto. As técnicas de comprobación da conformidade ofrecen a posibilidade de comparar o rexistro de eventos cun modelo de proceso que especifica este requisito, detectando os casos nos que o proceso de comprobación executouse nun punto incorrecto, ou mesmo casos nos que esta comprobación non se executou. Así, as técnicas de comprobación da conformidade poden utilizarse para detectar, localizar e explicar as desviacións, e para medir a súa gravidade. O terceiro tipo é a *Mellora*. O obxectivo da mellora é mellorar un modelo de proceso existente utilizando a información sobre o proceso rexistrada no rexistro de eventos. Un tipo de mellora é a *extensión*, que consiste en engadir nova información a un modelo de proceso baseado nun rexistro de eventos do mesmo proceso. Un exemplo de ampliación pode ser a adición de marcas de tempo ou datos de recursos ás actividades do modelo. Con esta información, pódense realizar outras análises, como buscar pescozos de botella ou a predición de tempo restante. Outro tipo de mellora é a *reparación*, na que o modelo se modifica para reflectir mellor realidade. Por exemplo, cambiar a estrutura de dúas actividades que foron modeladas como unha secuencia, pero que poden ser executadas en calquera orde. Mentres que a comprobación da conformidade identifica e describe as desviacións entre o modelo de proceso e o comportamento rexistrado, a mellora redeseña e amplía o modelo para aumentar a súa utilidade.

Como se pode observar, existe un elemento común entre todos os tipos nos que se clasifican as técnicas de minería de procesos, o *rexistro de eventos*. O rexistro de eventos almacena os datos rexistrados sobre o proceso que posteriormente son utilizados polas técnicas de minería de procesos. Para isto, cada *evento* —execución dunha *actividade*— rexistrado está relacionado con un *caso* —é dicir, a unha execución do proceso— e almacena unha marca de tempo que denota o momento no que se produciu o evento. Ademais, pódese almacenar outra información relacionada co evento, como o *recurso* que que o realizou. Con esta información, os eventos dun caso poden ser agrupados e ordenados pola de ocorrencia, formando unha secuencia de eventos pertencentes a esa instancia do proceso. A secuencia de acontecementos relacionados cun só caso tamén se denomina *traza*. Así, un rexistro de eventos é un conxunto de trazas que rexistra a execución de cada instancia do proceso.

Para realizar unha análise de minería de procesos, o requisito mínimo destes datos é que os eventos especifiquen: *i)* a execución do proceso á que pertencen *ii)* a actividade executada, e *iii)* o instante en que tiveron lugar. Con esta información, os eventos poden ser agrupados e ordenados para obter as trazas do proceso. Os eventos poden conter información adicional que pode enriquecer a análise do proceso. Por exemplo, os recursos implicados en cada actividade executada, o prezo do produto pedido, se o cliente contratou algún seguro, unha marca de tempo correspondente á hora de inicio de cada evento, etc. Ter en conta toda esta información ofrece a oportunidade de realizar análise de minería de procesos desde diferentes perspectivas. Non obstante, nesta Tese Doutoral centrámonos unicamente na perspectiva do comportamento, que se basa na análise das trazas coma secuencias de actividades.

Outro elemento común a todos os tipos de minería de procesos é o modelo de proceso. Un modelo de proceso é unha representación diagramática dun proceso baseada no comportamento almacenado no rexistro de eventos. O modelo de proceso é o elemento chave no que se centran os tres tipos de minería de procesos. Ademais, desempeña un papel preponderante na maioría das fases do ciclo de vida da BPM.

Normalmente, os modelos de procesos represéntanse utilizando formalismos como a Notación de Modelos de Procesos de Negocio (*Business Process Model Notation* ou BPMN en inglés), redes de Petri, redes de fluxo de traballo, etc. Nesta tese, representaremos a maioría dos modelos de procesos utilizando o formalismo de redes de Petri. Unha rede de Petri é un grafo dirixido composto por dous tipos de nodos: *lugares* e *transicións*, e onde os arcos conectan dous nodos de distinto tipo. Dise que unha transición está habilitada cando todos os lugares das súas entradas —cun arco cara a ela— conteñen, polo menos, un *token*. A execución dunha transición habilitada consome un token de cada lugar das súas entradas e xera un token en cada lugar das súas saídas —lugares cun arco desde a transición cara a eles—.

Un modelo de proceso representa o comportamento que está a ser, ou pode ser, executado no proceso. Pódese utilizar para inspeccionar o que está a suceder no proceso, ou para modelar o que pode suceder. A calidade dun modelo de proceso baséase en catro métricas: a *aptitude*, que cuantifica a medida en que o modelo descuberto pode reproducir con exactitude os casos rexistrados no rexistro de eventos; a *precisión*, que cuantifica a fracción do comportamento permitido polo modelo que non se observou no rexistro de eventos; a *xeneralización*, que avalía ata que punto o modelo resultante poderá reproducir futuros comportamentos non observados do proceso; e a *simplicidade*, que representa a complexidade estrutural do modelo.

Como xa se dixo, un modelo de proceso de baixa calidade pode comprometer o resto das análises e reducir significativamente as melloras que se poden realizar no proceso. Debido ao papel principal que desempeña o modelo de proceso nos ciclos de vida da BPM e PM, a construción dun modelo de proceso de alta calidade durante a fase de descubrimento é crucial para a análise xeral, co fin de mellorar o proceso na medida do posible.

Coa explosión de datos relacionados cos procesos que se rexistran hoxe en día, o descubrimento de modelos de procesos complexos volveuse máis común. Polo xeral, estes modelos de procesos complexos son descubertos a partir de procesos complexos nos que, para obter un bo nivel de aptitude, sacrifícanse a simplicidade e a precisión dos modelos de proceso. Nestes casos, o resultado é un modelo de proceso con pouca precisión e simplicidade, o que presenta diferentes problemas para as análises e melloras que se realizan nas outras etapas tanto de BPM como de PM: *i)* en canto á simplicidade, descubrir un modelo de proceso complexo, é dicir, dificilmente lexible de proceso, pode entorpecer totalmente a súa calidade dificultando a análise do proceso; *ii)* no caso da precisión, o modelo de proceso admite unha gran cantidade de comportamento non rexistrado no rexistro de eventos, o que reduce a confianza do modelo de proceso.

Con modelos de procesos complexos de baixa precisión, a detección de desviacións durante a conformidade e a mellora posterior vense moi dificultadas. Ademais, a maioría das etapas da BPM que dependen do modelo de proceso, como a análise, o redeseño e a implementación, tamén se ven obstaculizadas. Para facer fronte a esta carencia e mellorar as análises futuras, nos últimos anos desenvolvéronse moitas técnicas destinadas a obter modelos de procesos máis sinxelos e precisos. As estratexias seguidas por estas técnicas poden resumirse en dous grupos.

Por unha banda, a simplificación do rexistro de eventos: Algunhas técnicas céntranse na simplificación do rexistro de eventos para descubrir posteriormente un modelo de proceso máis sinxelo e preciso. O obxectivo é reter, no rexistro de eventos simplificado, o comportamento principal que soportará o novo modelo de proceso descuberto. Algunhas destas técnicas eliminan as trazas infrecuentes ou atípicas. Como unha traza infrecuente pode conter un comportamento frecuente —por exemplo, unha traza que rexistre o camiño máis frecuente, pero coa execución dunha actividade infrecuente no medio—, a eliminación de trazas completas pode eliminar tamén o comportamento frecuente. Para facer fronte a esta deficiencia, outras técnicas eliminan só os eventos que aparecen nun contexto infrecuente, ou todos os eventos das actividades que aparecen en contextos infrecuentes.

Por outra banda, a simplificación do modelo. Outras técnicas céntranse na simplificación directa do modelo de proceso. Para obter un modelo de proceso máis sinxelo e con maior precisión, e mellorar as análises futuras, estas técnicas aplican simplificacións estruturais ao modelo de proceso. Algunhas delas basean as simplificacións no comportamento rexistrado no rexistro, mentres outras realizan unha simplificación estrutural do modelo de proceso sen ningún outro coñecemento. Esta simplificación estrutural do modelo de proceso adoita producir modelos de proceso sen propiedades básicas de corrección, como a solidez (*soundness* en inglés), o que tamén dificulta futuras análises.

O obxectivo principal da simplificación do modelo de procesos é reducir a complexidade do modelo de proceso conservando a maior cantidade posible de comportamento. Así, para realizar unha boa simplificación, é desexable detectar o comportamento frecuente —é dicir, os subprocesos frecuentes— a reter. A maioría das técnicas comentadas non realizan unha procura do comportamento frecuente a reter. No seu lugar, simplifican o modelo de procesos eliminando o comportamento máis infrecuente e conservando o resto. Ademais, os enfoques seguidos para detectar o comportamento infrecuente céntranse só na frecuencia de cada actividade individual, ou de cada par de actividades —por exemplo, A seguida de B—, en lugar de detectar subprocesos. A simplificación realizada por estas técnicas podería mellorarse de dúas maneiras.

Por unha banda, para realizar unha mellor simplificación, poderíanse utilizar técnicas máis avanzadas de procura de comportamentos frecuentes ou infrecuentes. Como parte da minería de procesos, desenvolvéronse moitas técnicas centradas na extracción de información sobre o comportamento, como os subprocesos frecuentes. Debido á estrutura secuencial das trazas no rexistro de eventos, as primeiras técnicas utilizadas para este propósito foron as técnicas de minería de patróns secuenciais, recuperando secuencias de actividades que se observan frecuentemente no rexistro. Con todo, un proceso —ou un subproceso— pode conter estruturas máis complexas que as secuencias, como a concorrencia, as seleccións ou os bucles. Por esta razón, propuxéronse outras técnicas que superan este inconveniente para descubrir subprocesos frecuentes que soporten concorrencia, e mesmo seleccións e bucles.

Doutra banda, a eliminación do comportamento infrecuente que está a obstaculizar a simplicidade e a precisión do modelo de proceso pode alterar a estrutura dos subprocesos frecuentes. Por exemplo, consideremos un subproceso frecuente que contén unha parte infrecuente con moitos camiños entre as actividades `A` e `B`, no medio da súa estrutura. Ao eliminar esta parte infrecuente, a estrutura frecuente permanece, pero a información sobre o subproceso frecuente é inexacta, xa que o comportamento entre `A` e `B` non se representa. O modelo de proceso simplificado representa o subproceso frecuente coma se non se executase nada entre `A` e `B`. Isto pode levar a unha interpretación errónea do proceso, facendo que o analista pense que non está a ocorrer nada entre dúas actividades, cando en realidade podería haber un comportamento infrecuente —e potencialmente perigoso. Para evitar este problema de desinformación e, ao mesmo tempo, reducir a complexidade do proceso, o comportamento infrecuente podería encapsularse en actividades artificiais. Deste xeito, o comportamento que ofusca a visualización reduciríase a unha soa actividade —mellorando a simplicidade—, e seguiríase coñecendo a existencia e a localización exacta do comportamento infrecuente.

En resumo, nesta Tese Doutoral pretendemos facer fronte ás carencias relacionadas cos procesos complexos que impiden as análises futuras. Neste sentido, podemos identificar dous temas principais de interese. Por unha banda, interésanos a recuperación de subprocesos frecuentes e infrecuentes dos procesos complexos. Por outra banda, pretendemos utilizar esta información para simplificar o modelo de proceso descuberto mediante a abstracción do comportamento infrecuente, coa fin de obter un modelo de proceso máis sinxelo e preciso. Polo tanto, a motivación deste doutoramento pode resumirse en *"Simplificar modelos de proceso complexos utilizando de comportamento frecuentes e infrecuentes"*.

Coa fin de simplificar os modelos de proceso complexos mantendo a maior cantidade de comportamento posible, un dos obxectivos principais desta tese é detectar os subprocesos que se están executando máis frecuentemente. Estes subprocesos serán o comportamento que manteñamos na simplificación, maximizando desta forma a cantidade de comportamento modelado polo modelo de proceso simplificado. No campo da minería de procesos propuxéronse algunhas técnicas para a detección de estruturas frecuentes en procesos. Algunhas delas están deseñadas para descubrir estruturas simples como secuencias de actividades. Outras profundizan máis na complexidade destas estruturas, pero utilizan métodos para medir a súa frecuencia que poden levar a confusións nalgúns contextos. Para evitar estes problemas, e poder descubrir subprocesos con todo tipo de estruturas —secuencias, seleccións, concorrencia e bucles—, propoñemos un método de descubrimento de subprocesos frecuentes que utiliza a información do modelo de proceso. Desta forma, a primeira hipótese desta Tese Doutoral pode ser definida da seguinte forma: *O uso da información do modelo de proceso para descubrir subprocesos frecuentes reduce o espazo de procura e garante unha medición precisa da súa frecuencia.* Así, no Capítulo 2 descríbese WoMine, un algoritmo para descubrir subprocesos frecuentes dun modelo de proceso, medindo a súa frecuencia nas instancias do rexistro de eventos. WoMine descobre subprocesos frecuentes con todo tipo de estruturas — incluso ciclos de lonxitude n, estruturas moi comúns en procesos reais. O algoritmo realiza unha procura a-priori construíndo os subprocesos coas relacións no modelo de proceso, reducindo así o espazo de procura. Ademais, WoMine mide a frecuencia de cada subproceso tendo en conta as relacións do modelo de proceso, asegurando unha medición precisa e estrita da frecuencia. O algoritmo foi probado con 20 modelos de procesos sintéticos que van de 20 a 30 actividades únicas, e que conteñen secuencias, concorrencia, seleccións e bucles. Tamén se realizaron experimentos con 12 rexistros de eventos complexos reais publicados en competicións de Intelixencia de Procesos de Negocio.

De cara a detectar o comportamento infrecuente para abstraer na simplificación, é necesario descubrir os subprocesos que están ocorrendo infrecuentemente. Non hai moita investigación feita no eido do comportamento infrecuente. As técnicas propostas céntranse maiormente na detección de trazas infrecuentes, ou de actividades que se executan moi poucas veces. A problemática que xurde ao utilizar as trazas que se executan infrecuentemente é estas trazas poden conter comportamento frecuente á súa vez. Por exemplo, unha traza na que se rexistra a execución dunha actividade única en todo o rexistro de eventos vai ser infrecuente, pero o resto da traza pode ser frecuente. Isto obriga a usar máis técnicas para discernir do comportamento frecuente que almacena esa traza. Por este motivo, e co obxectivo de detectar únicamente os subprocesos infrecuentes, a segunda hipótese desta Tese Doutoral defínese como: *O uso da información do modelo de proceso para descubrir subprocesos infrecuentes reduce o espazo de procura e garante unha medición precisa da súa frecuencia.* Así, o Capítulo 3 describe WoMine-i, un algoritmo para detectar subprocesos infrecuentes dun modelo de proceso, medindo a súa frecuencia coas instancias do rexistro de eventos. A principal novidade do noso enfoque é que pode detectar subprocesos infrecuentes con todo tipo de estruturas —secuencias, seleccións, concorrencia e bucles. A capacidade de traballar con estas estruturas evita que WoMine-i interprete as trazas como secuencias de eventos. Ademais, a información extraída permite centrarse en subprocesos pouco frecuentes, e non analizar trazas completas pouco frecuentes. O algoritmo comparouse cualitativamente utilizando varios modelos de procesos sintéticos con todas as técnicas relacionadas, mostrando que o noso algoritmo atopa os patróns infrecuentes correctos e estima con precisión a súa frecuencia mentres que as técnicas relacionadas non o fan. Tamén se realizaron experimentos con rexistros de eventos reais de dous competicións de Intelixencia de Procesos de Negocio, *BPIC 2012* e *BPIC 2013*.

Por último, co comportamento frecuente e infrecuente identificados, proponse unha simplificacion do modelo de proceso para reducir a complexidade do mesmo, á vez que se mellora o equilibrio entre as medidas de aptitude e precisión. Para isto, a terceira hipótese desta Tese Doutoral defínese coma: *A simplificación dos procesos complexos mediante a abstracción do comportamento infrecuente en actividades artificiais produce modelos de procesos máis sinxelos cun mellor equilibrio entre aptitude e precisión.* Así, o Capítulo 4 describe dous algoritmos para a abstracción do comportamento nos modelos de procesos: UBeA e IBeA. UBeA é un algoritmo para abstraer o comportamento dun proceso en actividades artificiais utilizando as relacións entre as actividades, por tanto, tendo en conta estruturas como a concorrencia, as seleccións ou os bucles. A principal novidade de UBeA é que xera unha versión abstracta do proceso que describe o comportamento desexado, á vez que coñece a existencia e a localización exacta do comportamento abstraído. UBeA permite ao usuario especificar o comportamento a manter, é dicir, o comportamento desexado, o que o fai moi versátil. Por outra banda, IBeA é unha implementación específica de UBeA para simplificar os modelos de procesos mediante a abstracción do comportamento infrecuente, utilizando WoMine para detectar o comportamento desexado —considerando o comportamento infrecuente como non desexado— permitindo producir un modelo de proceso máis simple mentres se mantén un compromiso entre aptitude e precisión. IBeA foi validado cun conxunto de 11 rexistros de eventos complexos e reais, 10 deles do Business Process Intelligence Challenges (BPIC), e un do dominio da saúde. Os experimentos mostran que a simplificación de IBeA xera mellores modelos de procesos que outras técnicas de simplificación.

Finalmente, a tese remata co Capítulo 5, onde se presentan as conclusións sobre o traballo feito e se introducen posibles liñas de traballo futuro froito da investigación realizada.

Contents

CHAPTER 1

INTRODUCTION

“There are two kinds of people, those who do the work and those who take the credit. Try to be in the first group; there is less competition there."

— Indira Priyadarshini Gandhi

1.1 Motivation

Processes are everywhere. Every organization —either a non-profit organization, a governmental agency, or an enterprise— has to manage processes [15]. In order to improve these processes by increasing their performance, their behavioral analysis and study have become a must for every organization. As a result, the amount of recorded data related to the execution of these processes has been greatly increased during past years. This increase has created a need to develop techniques and methods in order to automate and improve the analysis of the recorded data.

In this context, Business Process Management (BPM) appears as a discipline offering a set of methods, techniques, and tools to identify, discover, analyze, redesign, execute, and monitor processes to optimize their performance [15]. In order to continuously improve the process, BPM defines a cyclic procedure formed by a set of stages to guide the analysis of the process by renourishing each stage with the information obtained in the previous ones. Figure 1.1 depicts this cyclic structure. The BPM lifecycle starts with the identification of the

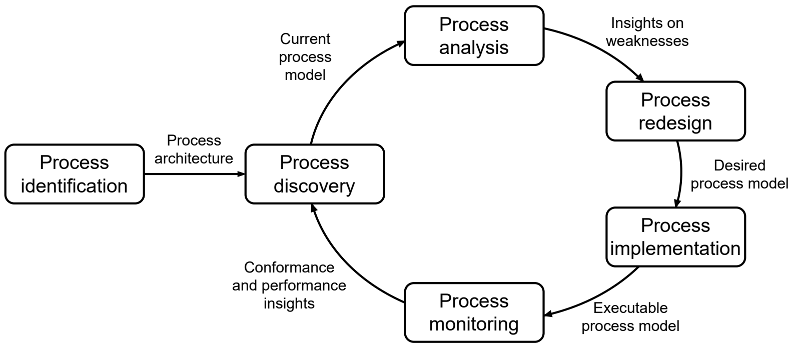


Figure 1.1: BPM lifecycle (adapted from (15)).

process, its architecture, and performance measures (*process identification*). Then, the current state of the process is documented (*process discovery*) to analyze it in the following stage in order to detect issues that are affecting the process performance (*process analysis*). The next stage consists of identifying changes to redesign the process and tackle the issues found in the previous stage (*process redesign*). These changes are implemented in the process as part of the next stage (*process implementation*). Finally, once the changes have been implemented, the last stage is to monitor the performance of the process (*process monitoring*) and start again the cycle to improve it in a continuous way.

Until recently, most of these stages had to be done manually, requiring a great amount of human interaction to design the model describing the behavior of the process, or to detect the issues to be improved. Process Mining (PM) has emerged as a research discipline to reduce the human interaction needed in these stages by increasing their automation. PM sits between machine learning and data mining on the one hand, and process modeling and analysis on the other hand, offering techniques to discover, monitor, and enhance business processes by extracting knowledge from event logs readily available in today's systems [45]. Figure 1.2 depicts how PM interacts with the real world and the elements of a process management system. As can be seen, process mining establishes a link between the recorded data and the process models describing the behavior of the process.

To establish this link, the techniques belonging to process mining are typically categorized into three main types (Figure 1.3):

- The first type is *Discovery* (Figure 1.3(a)). Discovery techniques take as input an event log with the recorded information about the process, and produce a process model de-

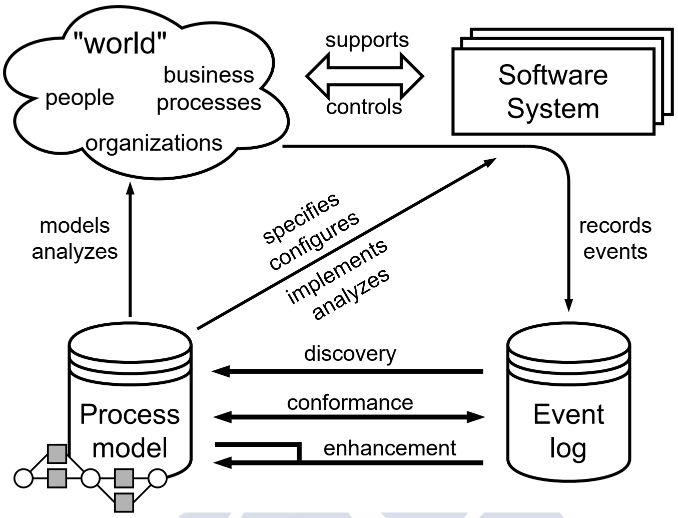
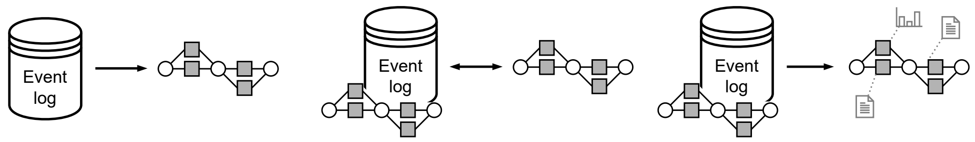


Figure 1.2: Process Mining framework (adapted from (45)).

scribing —fully or partially— the observed behavior. Although most of the discovery techniques do not take into account any a-priori information [3, 23, 49], some techniques require this knowledge to produce the process model [14, 19, 41].

- The second type is *Conformance* (Figure 1.3(b)). Conformance checking techniques compare an existing process model —either discovered or manually designed— with an event log of the same process [7, 21, 31]. These techniques are used to detect deviations between the (real) behavior recorded in the log and the behavior supported by the model. These deviations may hint at both issues in the process model that should be corrected, as well as incidents that have already occurred and must be avoided in the future. For example, take a manufacturing process where the quality of a product has to be checked at a specific point. Conformance checking techniques offer the possibility to compare the event log against a process model specifying this requirement, detecting the cases where the checking process has been executed at a wrong point, or even the cases where it has not been executed. Thus, conformance checking techniques may be used to detect, locate, and explain deviations, and to measure their severity [45].

- The third type is *Enhancement* (Figure 1.3(c)). The aim of enhancement is to improve



(a) Process discovery. (b) Conformance checking. (c) Process enhancement.

Figure 1.3: Process Mining types (adapted from (48)).

an existing process model using the information about the process recorded in the event log. One type of enhancement is *extension*, consisting of adding new information to a process model based on an event log of the same process. An example of extension can be the addition of timestamps or resource data to the activities of the model. With this information, further analyses like bottlenecks or time prediction can be performed [46]. Another type of enhancement is *repair*, where the model is modified to better reflect reality [2]. For example, changing the structure of two activities that were modeled as a sequence, but can be executed in any order. Whereas conformance checking identifies and describes the deviations between the process model and the recorded behavior, enhancement redesigns and extends the model to increase its utility.

Event log

There is a common element between all the types in which process mining techniques are classified: the *event log*. The event log stores the data recorded about the process which is later used by the process mining techniques. Table 1.1 shows a sample of an event log of a delivery company process where each row represents an *event*, i.e., the execution of an *activity*. As can be seen in this example, each event refers to a *case*, i.e., a process instance, and stores a *timestamp* denoting the instant at which the event took place. Furthermore, other information related to the event can be stored, such as the *resource* which performed it. With this information, the events of a case can be grouped and ordered by the timestamp of occurrence, forming a sequence of events belonging to that process instance. For example, considering case 512, first Fry *registered the order*, then he *checked the stock* to see if the product was available. Later, Bender *prepared and packed the order*. Finally, Leela *delivered the order* ending the process by *confirming the delivery*. The sequence of events related to a single case is also referred to as a *trace*. Thus, an event log is a set of traces recording the execution of each process instance.

Although the event log can be directly obtained in this example, it is common that the data stored in today's systems do not comply with the exact format shown in Table 1.1. Nevertheless, in most cases, this information can be easily extracted by applying preprocessing techniques to the raw data. In order to perform a process mining analysis, the minimum requirement for this data is that the events specify: *i)* the case instance to which they belong, *ii)* the executed activity, and *iii)* the instant in which they took place. With this information, the events can be grouped and ordered to obtain the traces of the process. As shown in the example, there may be additional information that can enrich the analysis of the process. For instance, the resources involved in each executed activity, the price of the ordered product, whether the customer took out any insurance, a timestamp corresponding to the start time of each event, etc. Taking into account all this information brings the opportunity to perform process mining analyses from different perspectives [27, 38, 45]. For example, four of the most common perspectives are:

Table 1.1: Sample of an event log of a fictional delivery company process, where each row corresponds to the execution of an activity, i.e., an event.

Case	***Activity***	***Timestamp***	***Resource***	...
⋮	⋮	⋮	⋮	⋮
512	*Register order*	*2020-03-15T09:10:01*	*Fry*	...
512	*Check stock*	*2020-03-15T09:15:51*	*Fry*	...
515	*Register order*	*2020-03-15T09:15:58*	*Zoidberg*	...
515	*Check stock*	*2020-03-15T09:20:25*	*Zoidberg*	...
515	*Purchase from supplier*	*2020-03-15T09:25:53*	*Zoidberg*	...
511	*Prepare and pack order*	*2020-03-15T11:32:36*	*Bender*	...
513	*Prepare and pack order*	*2020-03-15T11:40:34*	*Bender*	...
516	*Prepare and pack order*	*2020-03-15T11:50:48*	*Bender*	...
512	*Prepare and pack order*	*2020-03-15T11:56:27*	*Bender*	...
511	*Deliver order*	*2020-03-15T16:24:24*	*Amy*	...
513	*Deliver order*	*2020-03-15T16:24:24*	*Amy*	...
516	*Deliver order*	*2020-03-15T16:24:24*	*Leela*	...
512	*Deliver order*	*2020-03-15T16:24:24*	*Leela*	...
512	*Confirm delivery*	*2020-03-15T19:31:22*	*Leela*	...
⋮	⋮	⋮	⋮	⋮

- *Control-flow perspective*. This perspective, sometimes referred to as *behavioral perspective*, focuses on the order in which the activities have been, or should be, executed. The objective of the control-flow perspective is to represent the relations between the activities and the different paths of the process. As an example, *"The first activity must always be to register the order"*, or *"A purchase from the supplier must always be preceded by a stock checking"*. This perspective constitutes the foundation of a process model and is, usually, the starting point for a process mining analysis [27].

- *Resource perspective*. This perspective, sometimes referred to as *organizational perspective*, focuses on the information about the resources that execute each activity, and how they interact with each other. The resources of a process can be human or non-human —e.g., a machine required to execute an activity. The objectives of the resource perspective are, among others, to analyze the structure of the organization by constructing social network graphs, and to detect or establish resource constraints of the process. As an example, *"The person who performs the purchase from the supplier must be the same as the person who checked the stock"*, or *"The delivery of the order must be performed by a courier"*.

- *Time perspective*. This perspective focuses on the time-related aspects of the process. In addition to the time measures followed to order the events of a trace, the execution of the activities of a process takes time, and there is usually time between the execution of an activity and the execution of the next one. The objective of the time perspective is to analyze these aspects to discover bottlenecks, to predict the remaining time of running activities or cases, to detect blank time intervals between activities, etc. As an example, *"The preparing and packing of an order must not take more than 20 minutes"*, or *"On average, the orders of May have been processed and prepared in 7 hours and shipped in 5 hours"*.

- *Data perspective*. This perspective, sometimes referred to as *informational perspective*, focuses on the transmission of data between the activities of the process. The objective of the data perspective is to detect or establish restrictions and control-flow decisions w.r.t. the information related to the activities of the process. As an example, *"If there are no units of the ordered product in stock, purchase it from supplier"*, or *"If the order is insured, perform the delivery in person"*.

Although all perspectives provide useful information to the analysis, the control-flow perspective is the most widely adopted. From the control-flow point of view, a trace is just a sequence of activities ordered by the timestamp of their execution. For example, trace 512 can be written as ⟨ *Register order*, *Check stock*, *Prepare and pack order*, *Deliver order*, *Confirm delivery* ⟩.

It must be noted that these perspectives are not exclusive. It is common to combine them in order to enrich the information and to obtain more useful insights about the process. For example, the prediction of the next activity to be executed can be performed based on the control-flow information, but also taking into account the resource and time information of each event [8, 17].

Process model

Another common element in all PM types is the process model. A process model is a diagrammatic representation of a process based on the behavior recorded in the event log. The process model is the key element in which the three types of PM are centered (c.f. Figure 1.3). Furthermore, it plays a dominant role in most of the BPM lifecycle stages [45], as can be seen in Figure 1.1.

Typically, process models are represented using formalisms such as the Business Process Model Notation (BPMN), Petri nets, Workflow nets, etc. In this thesis, we will depict most of the process models using the Petri net formalism. A Petri net [13, 34] is a directed graph composed of two types of nodes: places and transitions —circles and boxes, respectively—, and where the arcs connect two nodes of different types. A transition is said to be *enabled* when all the places of its inputs —with an arc to it— contain, at least, a token —represented by a dot. The execution of an enabled transition consumes a token from each place of its inputs and generates a token in each place of its outputs —places with an arc from the transition to them.

Figure 1.4 shows an example of a process model (Petri net) corresponding to the behavior recorded in the event log of Table 1.1. As can be seen, a process model depicts the behavior that is being, or can be, executed in the process. It can be used to inspect what is happening in the process, or to model what can happen. In this simple example, the trace 512 is supported through the execution of the *Register order*, which enables *Check Stock*; this one enables both *Purchase from supplier* and *Prepare and pack order*, being the latter the executed one, followed by *Delivery order* and *Confirm delivery*.

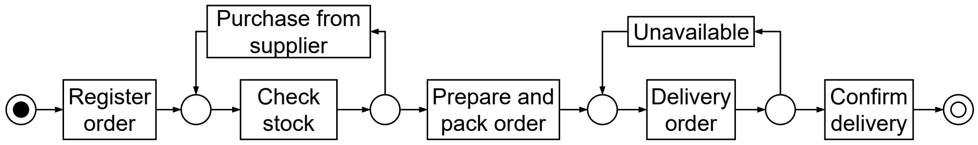


Figure 1.4: Example of a process model (using the Petri net formalism) corresponding to the event log in 1.1.

The quality of a process model is based on four metrics: *fitness replay*, which quantifies the extent to which the discovered model can accurately reproduce the cases recorded in the log; *precision*, which quantifies the fraction of the behavior allowed by the model which is not seen in the event log; *generalization*, which assesses the extent to which the resulting model will be able to reproduce future unobserved behavior of the process; and *simplicity*, which represents the structural complexity of the model [6].

As said before, a low-quality process model can compromise the rest of the analyses and reduce significantly the improvements that can be done to the process. Due to the main role that the process model plays in both the BPM and PM lifecycles, building a high-quality process model during the discovery phase is crucial to the overall analysis, in order to improve the process as much as possible.

The need of process model simplification

With the explosion of process-related data being recorded nowadays, the discovery of complex process models has become more common. Usually, these complex process models are discovered from complex processes where, in order to obtain a good level of fitness replay, the simplicity and precision of the process models are sacrificed. In these cases, the result is a process model with low precision and simplicity, presenting different problems for the analyses and improvements performed in the other stages of both BPM and PM:

- Regarding simplicity, discovering a complex process model, i.e., a hardly readable process model, can totally hinder its quality [7] making difficult the analysis of the process.

- In the case of precision, the process model supports a great amount of behavior not recorded in the event log, which reduces the confidence of the process model.

As an example, Figure 1.5 depicts the process model of the trajectories of patients treated for aortic stenosis in a Spanish hospital. As can be seen, little information about the process

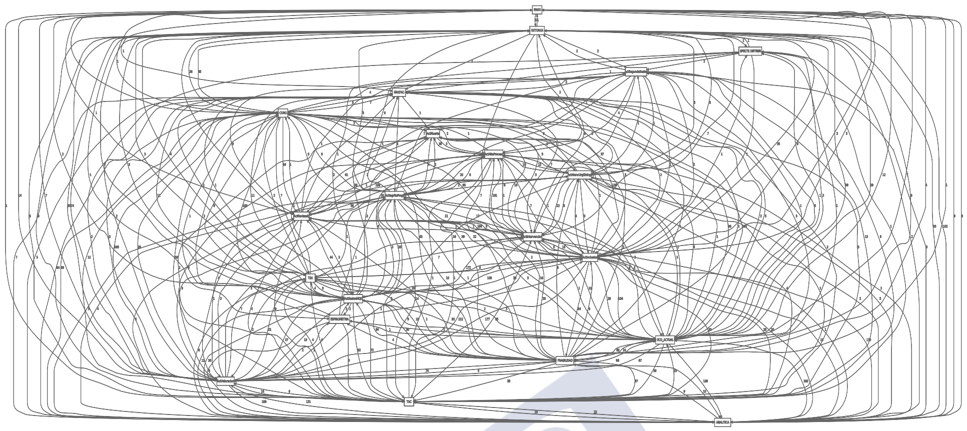


Figure 1.5: Complex process model (spaghetti model) of an aortic stenosis process in a Spanish hospital.

can be extracted from such a complex structure. Furthermore, the low precision value in this process model makes it difficult to know which supported behavior is happening in the process, and which is not.

With complex process models of low precision, the detection of deviations during conformance, and the posterior enhancement are greatly hindered. Furthermore, most stages of BPM depending on the process model, such as the analysis, redesign, and implementation, are also hindered. To cope with this shortcoming and improve future analyses, many techniques have been developed in the past years aiming to obtain simpler and more precise process models. Nevertheless, most of the existent proposals for process model simplification rely on simple statistics such as the individual frequency of each activity, or the individual frequency of each directly-follows relation —i.e. the frequency of apparition of activity *B* following activity *A*. Furthermore, these methods propose to remove the detected infrequent behavior, either full traces, single events or activities, which may be harmful to further analyses in some contexts.

1.2 Hypothesis

We are interested in the simplification of complex process models maintaining support for as much behavior of the process as possible, while increasing the precision of the process model. In order to retain as much behavior as possible, we propose to detect the subprocesses that are being executed more frequently. On the other hand, to increase the precision, we propose to avoid the support of the process model for the subprocesses that are observed fewer times. The support for these infrequent subprocesses often increases the amount of unobserved behavior modeled by the process model, decreasing the precision. In this way, the simplification reduces the behavior that is harming the most to the precision of the process model.

Hence, the hypothesis this Ph.D. Thesis addresses is:

The simplification of complex process models through the detection of frequent and infrequent subprocesses generates simpler process models with a good trade-off between fitness and precision.

1.3 Objectives

The main objective of this Ph.D. Thesis is to simplify process models by avoiding the removal of behavior, in order to maintain the integrity of the information the process model provides, while producing a simpler process model with a good trade-off between fitness and precision. To achieve this, three specific objectives have been pursued:

O1. The extraction of frequent subprocesses.

The first objective of this Ph.D. Thesis is to discover subprocesses from a given process model, which are executed frequently in the event log. The discovered subprocesses must be supported by the process model —i.e. the subprocess must be a subgraph of the process model— and their frequency must be precisely measured. This objective has been achieved in the following publication:

D. Chapela-Campa, M. Mucientes, and M. Lama. Mining frequent patterns in process models. *Information Sciences*, 472:235–257, 2019. (DOI 10.1016/j.ins.2018.09.011)

O2. The extraction of infrequent subprocesses.

The second objective of this Ph.D. Thesis is to discover subprocesses from a given process model, which are executed infrequently in the event log. The discovered subprocesses must be supported by the process model and their frequency must be precisely measured. This objective has been achieved in the following publication:

D. Chapela-Campa, M. Mucientes, and M. Lama. Discovering Infrequent Behavioral Patterns in Process Models. In *15th International Conference on Business Process Management (BPM 2017)*, volume 10445 of *Lecture Notes in Computer Science*, pages 324–340, Springer, 2017.
(DOI 10.1007/978-3-319-65000-5_19).

O3. The simplification of process models by abstracting the infrequent behavior.

The third objective of this Ph.D. Thesis is to simplify a process model by abstracting the infrequent behavior, while retaining the frequent one. For this, the use of the frequent and infrequent subprocesses from objectives O1 and O2 should be considered, in order to abstract into artificial activities the infrequent behavior of the process. This objective has been achieved in the following publication:

D. Chapela-Campa, M. Mucientes, and M. Lama. Understanding complex process models by abstracting infrequent behavior. *Future Generation Computer Systems*, 113:428–440, 2020.
(DOI 10.1016/j.future.2020.07.030)

1.4 Methodology

This Ph.D. Thesis follows the scientific method, an iterative process composed of a first phase to specify the objectives to pursue, followed by a study of the state of the art techniques, an information and data search, the development of the solution, and its validation. This method is followed in the process to achieve each of the objectives of this dissertation. It must be noted that the schedule of the different phases is considered flexible, as one phase can start before the previous one finishes —e.g. start to develop a solution before the data gathering ends. Furthermore, the result of one of the phases can cause a restart of the iterative process to perform adjustments —e.g. modifications in the solution due to poor performance in the validation results. Each of the different phases of this method is applied in this dissertation as follows:

- **Objective specification**. This phase consists of the description and specification of the objectives designed to validate the hypothesis under consideration.

- **State of the art**. Analyze the strengths and weaknesses of the existent approaches that address the proposed objectives. Study and evaluate other presented approaches that may be useful to solve the pursued objectives. Some methodological tools used in this phase are the analysis of case studies that address some of the proposed objectives, and publications describing techniques with the same purpose.

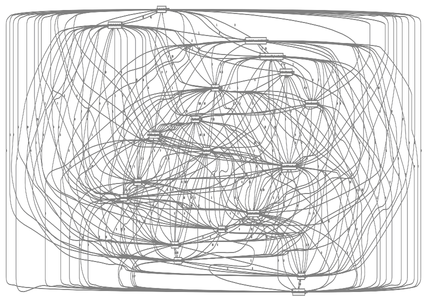
- **Information and data gathering**. Search for data in order to test the proposed solution while it is being developed, and to perform the final validation. The main tool to obtain this information is to perform a search for process event logs in public repositories and process mining publications. Nevertheless, other methodological tools such as obtaining the data from observational databases of associate companies can be considered.

- **Development of a solution**. Design and implement an algorithm (or algorithms) to solve the problems defined in the objectives. The development of the algorithms is performed by following the implementation and test methodology, consisting of implementing modularly the different features of the algorithm and testing them individually before validating the complete approach.

- **Validation**. This phase consists of the validation of the complete proposed solution. This empirical evaluation is performed by using the data gathered in previous phases, and analyzing the obtained results. In case that the quality of the results is not adequate, the process returns to previous phases to perform the modifications needed until those results are satisfactory.

1.5 Discussion

The analysis of processes, either by Business Process Management (BPM) or Process Mining (PM) techniques, has become a must for every organization in order to improve their performance. As we have already discussed in Sect. 1.1, the role of the process model is crucial in most of the BPM and PM phases. During past years, the amount of process-related data that has been gathered by information systems has greatly increased. With more information and behavior related to the processes being recorded, the apparition of complex processes and complex process models —with hundreds of edges and activities— has become more common. Figure 1.6 depicts two examples of complex process models of an e-learning process from a Spanish university (Figure 1.6(a)) and a travel permit process in a Dutch university


(a) Spaghetti process model of an event log recording the interaction of students from a Spanish university with a media visualization tool [37].

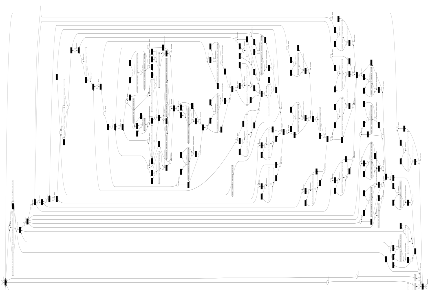

(b) Sample of a spaghetti process model of an event log recording the travel permit process in a university [47].

Figure 1.6: Two examples of spaghetti process models corresponding to real processes.

(Figure 1.6(b)). The complexity of these discovered process models —characterized by low simplicity and precision— hinder the detection of deviations during conformance, and the posterior enhancement. Furthermore, most stages of BPM depending on the process model, such as the analysis, redesign, and implementation, are also hindered. For this reason, the simplification of complex process models is a promising research field that can help the analysis of complex processes.

In order to cope with the shortcomings related to complex process models, and improve future analyses, many techniques have been developed in the past years aiming to obtain simpler and more precise process models. The strategies followed by these techniques can be summarized in two groups:

- *Model simplification*. Some techniques focus on the direct simplification of the discovered process model. In order to obtain a simpler process model with higher precision, and improve future analyses, these techniques apply structural simplifications to the process model. Some of them base the simplifications on the behavior recorded in the event log [12], and others perform a structural simplification of the process model without any further knowledge [16]. In [16], an approach to simplify discovered process models while controlling the precision and generalization is presented. The process model, expressed in terms of a Petri net, is unfolded into a branching process using the event log, then filtered retaining the frequent parts, and finally folded again into a simpler process model capturing the desired behavior. Other approaches focusing on fitness and precision are, for instance, the collection of event log-based techniques presented in [12]. They first rank the importance of the model places and arcs using the event log, and then simplify with different alternatives maintaining the more important arcs and places. Nevertheless, these techniques often sacrifice some correctness properties of the process model such as soundness, which also hinders future analyses, in order to maintain a good trade-off between fitness and precision.

- *Event log simplification*. Other techniques focus on simplifying the event log in order to later discover a simpler and more precise process model by using a discovery algorithm. The objective is to retain, in the simplified event log, the main behavior that the simpler discovered process model will support. Some of these techniques focus on the removal of infrequent or outlier traces [9, 11, 26, 39]. For example, in [39], authors identify and remove outlier traces using the probability of occurrence of each event conditioned by both its k predecessors and its k successors. This allows to identify the events with a low probability of occurrence, based on its surrounding behavior —i.e. how probable is that an activity follows, or is followed by, a sequence of activities.

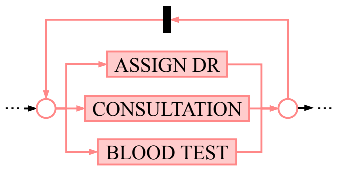
  Instead of removing full traces, the removal of the events which appear in an infrequent context is also proposed in [39]. Following this idea, in [44], a set of techniques that remove all the events of the activities with high entropy are presented. These techniques assign an entropy to each activity depending on their distribution of occurrence in the event log —i.e., based on the directly-precedes and directly-follows relations among the activities—, and remove the most chaotic activities from the event log in order to simplify it.

  Another alternative, instead of removing behavior, is to abstract subprocesses in the event log by replacing the execution of multiple activities with one [28, 29]. The key point of these techniques is to choose which subprocesses to abstract in order not to lose too much behavior w.r.t. the original process.

Many of the existent proposals for process model simplification rely on simple statistics such as the individual frequency of each activity, or the individual frequency of each directly-follows relation —i.e. the frequency of apparition of activity B following activity A. Furthermore, these methods propose to remove the detected infrequent behavior, either full traces, single events, or activities. Nevertheless, as it has been said, the removal of behavior from the process model can lead to misconclusions in some contexts. For example, consider

Trace	
... ASSIGN DR – CONSULTATION – BLOOD TEST ...	(×21)
... ASSIGN DR – BLOOD TEST – CONSULTATION ...	(×27)
... BLOOD TEST – ASSIGN DR – ASSIGN DR ...	(×4)
... CONSULTATION – BLOOD TEST ...	(×7)
... ASSIGN DR – CONSULTATION – ASSIGN DR ...	(×5)
...	

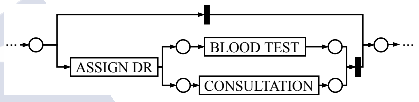
(a) Partial event log.



(b) Process model of the event log in Figure 1.7(a).

Trace	
... ASSIGN DR – CONSULTATION – BLOOD TEST ...	(×21)
... ASSIGN DR – BLOOD TEST – CONSULTATION ...	(×27)
... ...	

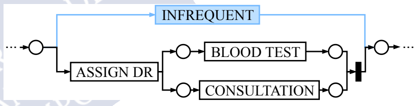
(c) Event log simplified by removing the infrequent behavior.



(d) Process model of the event log in Figure 1.7(c).

Trace	
... ASSIGN DR – CONSULTATION – BLOOD TEST ...	(×21)
... ASSIGN DR – BLOOD TEST – CONSULTATION ...	(×27)
... INFREQUENT ...	

(e) Event log simplified by abstracting the infrequent behavior.



(f) Process model of the event log in Figure 1.7(e).

Figure 1.7: Motivational example to show the interest of behavior abstraction.

the subprocess depicted in Figure 1.7. This figure depicts a sample of a log concerning 3 activities occurring in many orders in Figure 1.7(a), with its corresponding process model in Figure 1.7(b). In this type of scenario, where all the activities occur in almost any order, it is usual to obtain a flower-like structure such as the one depicted in Figure 1.7(b). But, in some cases, there is a latent structure hindered by the infrequent behavior. As can be seen, the common behavior is to first assign a doctor, and then perform a blood test either before or after going to the consulting room. This behavior cannot be observed due to the atypical cases hindering the visualization.

The removal of behavior, either full traces, activities, or events may be a solution, but it presents some drawbacks:

- **Removal of full traces.** The complete removal of traces also removes the frequent behavior they may contain, which might be important for the analysis of the process. For example, the traces containing the infrequent behavior in Figure 1.7(a) might contain frequent subprocesses in previous or future parts of the trace. Furthermore, there are complex processes with high variability in the event log where each trace follows a different activity sequence and, hence, all traces has an infrequent part —either a single event or an infrequent subprocess. Although these traces might contain frequent subprocesses, all of them would be removed due to their infrequent parts.

- **Removal of activities.** To avoid this loss of frequent behavior, an alternative to the removal of the complete infrequent traces is to detect the activities corresponding to the infrequent behavior and remove all the events related to them. Nevertheless, this can cause the removal of events in both infrequent and frequent contexts. For example, as can be seen in Figure 1.7(a), the infrequent behavior —i.e. the highlighted events— that is obfuscating the visualization of the frequent subprocess is composed of the same activities and, hence, the removal of all the events corresponding to those activities will remove also the frequent subprocess.

- **Removal of events.** In order to maintain the frequent subprocess and the frequent behavior in the infrequent traces, some techniques remove only the events corresponding to the infrequent behavior —not all the events of an activity, but only the infrequent ones. However, a drawback of this removal is that it generates inexistent paths in the process —each removal creates an immediate path from the previous events to the succeeding ones. For example, consider a frequent subprocess containing an infrequent part with many paths between activities A and B, in the middle of its structure. By removing this infrequent part the frequent structure remains, but the information about the frequent subprocess is inaccurate, as the behavior between A and B is not depicted. The simplified process model depicts the frequent subprocess as if nothing were executed between A and B. This can lead to a misinterpretation of the process, making the analyst think that nothing is happening between two activities, when actually there might be infrequent —and potentially harmful— behavior occurring. In the example in Figure 1.7(c) and Figure 1.7(d), the removal of the infrequent behavior produces an artificial path allowing to skip the execution of the depicted activities. This result may be sufficient if the analyst is aware of this, but it can lead to a misinterpretation of the process. Furthermore, the analyst cannot further be sure if any connection between two activities is trustworthy, or if the connection is replacing any removed infrequent behavior.

To cope with these shortcomings and maintain the integrity of the relations depicted by the process model, the alternative we propose in this Ph.D. Thesis is to perform a simplification by abstracting the infrequent behavior. Figure 1.7(e) shows an example of this abstraction, where the infrequent behavior is encapsulated into artificial activities —e.g. activity `INFREQUENT`—, obtaining the structure shown in Figure 1.7(f). This abstraction reduces the complexity of the process model on the one hand, while does not entirely remove the infrequent behavior from it on the other hand. The frequent subprocesses can be easily identified, and the artificial activities abstracting the infrequent behavior can even store the abstracted subtraces to show, if the analyst requests it, the encapsulated behavior.

The main objective of process model simplification is to reduce the complexity of the process model retaining as much behavior as possible. Thus, in order to perform a good simplification by abstracting the infrequent behavior, it is desirable to detect the frequent behavior —i.e. the frequent subprocesses— to retain. Most of the commented techniques do not perform a search for the frequent behavior to retain. Instead, they simplify the process model by identifying the most infrequent behavior, and removing the traces or events related to it. Furthermore, the approaches followed to detect the infrequent behavior often focus only on the frequency of single activities, or pairs of activities —e.g. *A* followed by *B*, instead of detecting subprocesses—, and do not take into account relations such as concurrency. With the abstraction of the infrequent behavior, we aim to improve the simplification performed by these techniques in two ways:

- In order to perform a better simplification, more advanced techniques to search for frequent or infrequent behavior could be used. As part of process mining, many techniques have been developed focusing on the extraction of behavioral information such as frequent subprocesses. Due to the sequential structure of the traces in the event log, the first techniques used for this purpose were sequential pattern mining techniques [1, 33], retrieving sequences of activities that are frequently observed in the log. Nevertheless, a process —or a subprocess— can contain more complex structures than sequences, like concurrency, choices, or loops. For this reason, other techniques overcoming this drawback were proposed for discovering frequent subprocesses supporting concurrency [22, 30], and even choices and loops [42].

- The removal of the infrequent behavior from the process model may not be desirable, as it has been stated. To avoid related misinformation problems, and to reduce the complexity of the process at the same time, the infrequent behavior can be encapsulated into artificial activities. In this way, the behavior obfuscating the visualization would be reduced to a single activity —improving the simplicity—, and its existence and exact location would still be known.

To perform this abstraction, the frequent behavior to maintain and the infrequent behavior to abstract must be identified. We want this behavior to be composed of subprocesses, and not by single relations or activities, in order to increase the quality of the retained behavior. For this reason, we first propose two algorithms to discover frequent and infrequent subprocesses. Then, once the behavior to abstract is known, we propose a technique to perform the abstraction of the infrequent behavior by replacing it with artificial activities. The following sections discuss in more detail each one of the proposed algorithms and their contributions.

Frequent behavior extraction

As it has been stated, the extraction of frequent behavior can be crucial in complex processes where the discovered process model presents low simplicity and precision values. In this context, the frequency of a subprocess can be measured in two ways: *i)* the absolute number of executions [22, 42], e.g. a structure executed 10,000 times in the event log; or *ii)* the relative number of executions w.r.t. the traces in the event log [5, 20], e.g. a subprocess observed the 80% of the times the process is executed. In any case, the quality of the simplification depends highly on the identification of the frequent subprocesses to maintain, and retaining the more frequent subprocesses during the simplification is useful for future analyses. For example, focusing an improvement on a frequent subprocess will likely return more benefits, as this improved part of the process is being executed many times. Furthermore, a simplified process model maintaining the frequent subprocesses describes better what is happening in the process, as it supports more behavior.

Many techniques have been developed focusing on the extraction of frequent behavior:

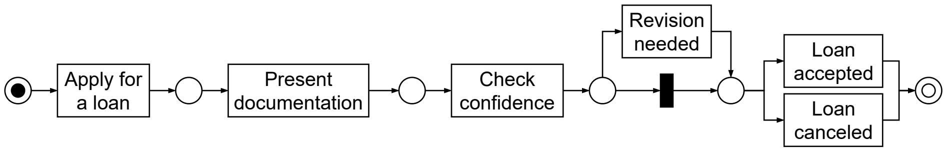
- Sequential pattern mining (SPM) algorithms were the first techniques used for frequent behavior extraction [1, 33]. SPM techniques retrieve sequences of elements that appear frequently in a given set of sequences. Thus, by applying these techniques to the sequences of activities that form the event log —from a control-flow perspective—, frequent sequential subprocesses can be extracted. One of the main drawbacks of using sequential pattern mining for this purpose is that it cannot detect concurrency. A subprocess can be formed by two activities (e.g. A and B) that can be executed concurrently and, thus, appear interleaved in the event log (A,B and B,A). SPM techniques consider each order as a different sequence, and thus, divide the real frequency of the subprocess.

- To cope with this shortcoming, episode mining techniques perform a similar search supporting concurrency [22, 30]. An episode is a collection of activities occurring close to each other —within a predefined *window* size. In this way, episode mining algorithms retrieve sets of activities that are frequently executed together without the restriction of having always the same order and, hence, supporting interleaving. One of the drawbacks of episode mining techniques is that they do not support loops nor choices —i.e. a split in the process where only one branch can be executed.

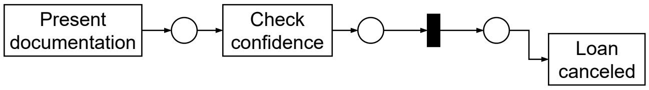
- Another technique to search for frequent subprocesses supporting concurrency is *w*-find [20]. This approach uses the process model to build the subprocesses, reducing the search space. *w*-find performs an a-priori search expanding the frequent subprocesses through the connections depicted in the process model. Nevertheless, similar to episode mining techniques, the frequent subprocesses cannot contain selections nor loops.

- Finally, the discovery of local process models [42] adds support for choices and loops by discovering small process models representing subparts of the traces, instead of a complex process model representing the behavior of full traces. The evaluation of the frequency is done with an alignment-based method which, starting with an initial marking, considers that the model is executed when the final marking is reached. A drawback of this technique is the high runtime due to the large search space. To build the local process models, this technique combines all the activities of the process among them, and expands them until the local process model becomes infrequent —an a-priori search. To cope with this, other improvements reducing the search space have also been published [10, 43].

Previous techniques evaluate the frequency of the subprocesses by checking their execution in the traces of the event log. They follow different strategies for that checking, but a common characteristic is present in all of them: they discard the execution of activities



(a) Example of a simple process.



(b) Example of a sequential subprocess.

Figure 1.8: Process example to show the problems related with the frequency checking of a subprocess.

not present in the subprocess. From the control-flow point of view, this might cause a false positive in the execution of the subprocess in some cases. For example, consider the simple process depicted in Figure 1.8(a), and the following trace (*trace 1*): ⟨ *Apply for a loan*, *Present documentation*, *Check confidence*, *Revision needed*, *Loan canceled* ⟩. In this case, a client applies for a loan and presents the documentation, the company checks the confidence of the application to make a decision, a revision of the case is needed, the revision is not performed in a fixed number of days and, thus, the loan is automatically canceled. Now consider the following trace (*trace 2*): ⟨ *Apply for a loan*, *Present documentation*, *Check confidence*, *Loan canceled* ⟩. In *trace 2* the same procedure is followed, but there is no need for a revision and the loan is canceled as a result of the checking process (e.g. a negative result). By checking the frequency of a subprocess based only on its activities —as commented techniques do—, the sequential subprocess depicted in Figure 1.8(b) would be detected as executed in both traces. This could lead the analyst to make a wrong conclusion, as there is a subset of the traces where activity *Revision needed* is being executed before the cancellation of the loan, but it is not being detected.

Regardless of this, the execution of a subprocess must not be discarded due to the execution, in the middle, of external activities —the external activity might belong to a concurrent branch unrelated to the subprocess. Concurrency is a common structure in processes, but, as explained before, external activities might be truly disrupting the execution of the subprocess in some cases. To cope with this problem, previous techniques could be adapted to support explicit restrictions, and let the analyst declare that if a determined activity is executed in the middle of a subprocess, the subprocess is not strictly being executed. Nevertheless, this would require the interaction and supervision of human resources. Another option is to use the information automatically extracted by discovery algorithms. A discovery algorithm detects the relations between the activities of the process. These relations can be used in the subprocess discovery to ensure that a given subprocess is being executed without any disruption. For this reason, we propose to use the relations in the process model in the search for frequent subprocesses.

The first contribution of this Ph.D. Thesis is WoMine, an algorithm to discover the frequent maximal subprocesses of a process model, supporting all types of structures —even n-length cycles—, and ensuring their frequency w.r.t. the traces of the event log. WoMine performs an a-priori search in the process model for the subprocesses —i.e. substructures of the process model— that are observed in a percentage of traces in the event log considered frequent. To check if a subprocess has been executed in a trace, WoMine performs a replay of the trace in the process model taking into account all the activities of the process, and ensuring that the subprocess is correctly executed.

The main contributions of this proposal are as follows:

- The proposed a-priori search allows to discover subprocesses with all types of structures and reduces the search space in two ways. First, by using the relations depicted in the process model, the search space is reduced to the subprocesses supported by the process model —i.e. all substructures of the process model. For example, if the process model does not depict a relation between two activities —e.g. activity A is sometimes followed by activity B—, this relation will not be part of any subprocess in the search space. And second, as any a-priori search, it takes advantage of the monotonic property of the frequency of a subprocess. The search starts with the smaller subprocesses of the process model, and increases their size by adding activities and relations from the process model, until the subprocess becomes infrequent. In this way, the search space is pruned when a subprocess is detected as infrequent, as all the subprocesses containing it will be infrequent too.

- To ensure a strict measurement of the subprocesses frequency, we propose to perform a replay of the trace in the process model taking into account all the activities, and analyzing if the subprocess is being correctly executed or not. In this way, we ensure a strict and precise measurement of the frequency of the subprocesses.

- The a-priori search retains all the frequent subprocesses found through the expansion process. For this reason, the result is a set with subprocesses modeling behavior already modeled by other subprocesses. To reduce the redundancy in the results, we propose a postprocessing stage to retain only the maximal frequent subprocesses. A subprocess in a collection is considered maximal when the behavior it models is not contained in any other subprocess of the collection. Hence, the postprocessing stage compares the obtained frequent subprocesses, and retains only those subprocesses that model behavior that is not contained into others.

Infrequent behavior extraction

As commented in previous sections, the search for frequent behavior plays a crucial role in the analysis of complex processes. Nevertheless, the discovery of infrequent subprocesses can be also interesting in order to simplify a complex process model. To obtain a simple and precise process model, but supporting as much behavior from the event log as possible, it is desirable to maintain the frequent behavior and, thus, abstract the infrequent one. The detection of infrequent subprocesses can be combined with the detection of frequent subprocesses in order to optimize the simplification.

Furthermore, the analysis of infrequent behavior can be useful on its own. There are scenarios where an infrequent subprocess can hint at erroneous behavior which must be examined. For instance, in insurance companies, infrequent behavior can be used to recognize fraudulent claims [50]. It can be also useful to detect intrusions in networks [32], or failures in software behavior [25]. Additionally, in well-structured processes, the behavior supported by a model is designed and expected to be executed. A substructure of the model with a low frequency of execution can hint at a path in the process that must be reinforced in order to increase its frequency or, conversely, where the underused assigned resources could be restructured to optimize the process.

Few techniques have been developed focusing on infrequent behavior. In [9], a state automaton with each state representing an activity of the log is built. A valuated arc between two states is added when one of them is followed by the other one in the log. Its value increases as this relation appears in the log. Afterward, the infrequent arcs are used to filter infrequent traces. The technique used by Lu et al. in [26] also filters full traces using the infrequent parts of a process model. In this case, models are built by merging the behavior in a subset of traces. Bezerra et al. [11] search for infrequent or anomalous traces in the log analyzing the whole trace. They present three approaches to filter infrequent traces depending on their frequency and conformance.

A drawback of most of these techniques is that they identify the infrequent behavior based on the absolute (individual) frequency of each relation. An infrequent subprocess can contain relations that, as part of other subprocesses, are executed frequently. Furthermore, all these techniques focus on the detection of infrequent traces, instead of infrequent subprocesses. Their result is a set of "outlier" or infrequent traces. Usually, a trace is infrequent because a part of it —i.e. a subtrace— is infrequent, but most of the behavior it records is part of frequent subprocesses. For instance, consider again *trace 1*: ⟨ *Apply for a loan*, *Present documentation*, *Check confidence*, *Revision needed*, *Loan canceled* ⟩ of the process in Figure 1.8(a). The subprocess consisting of checking the confidence, needing a revision, and canceling the loan might be infrequent —it may only appear in a small percentage of the event log traces. Nevertheless, the start of the trace (⟨ *Apply for a loan*, *Present documentation*, *Check confidence* ⟩) is a frequent subprocess being executed in all traces. Detecting *trace 1* as infrequent or outlier may be correct, but without further analyses, it is impossible to know which parts of the traces are infrequent, and which are not.

The second contribution of this Ph.D. Thesis is WoMine-i, an algorithm to discover the infrequent subprocesses of a process model, supporting all types of structures —even n-length cycles—, and ensuring their frequency w.r.t. the traces of the event log. WoMine-i adapts the pruning strategy followed by the a-priori search performed by WoMine, as well as the postprocessing stage in order to search for the infrequent subprocesses, instead of the frequent ones. This algorithm takes advantage of the frequency measurement method of WoMine to ensure in the same way that the frequency of the subprocesses is strictly measured. Finally, due to the monotonic nature of the frequency of a subprocess explained in the previous section, WoMine-i retrieves the minimal infrequent subprocesses.

The main contributions of this proposal are as follows:

- In the same way as WoMine, one of the contributions of this algorithm relies on the search space pruning, which is performed in two ways. First, the relations depicted in the process model are used to build the subprocesses. This reduces the search space to the subprocesses supported by the process model —i.e. all substructures of the process model. And second, the search starts with the smaller subprocesses of the process model, and expands them by adding activities and relations from the process model until the subprocesses are no longer promising.

- We propose to take advantage of the method to measure the frequency of a subprocess presented in WoMine. In this way, to analyze if the subprocess is being correctly executed or not, a replay of each trace in the process model is performed. In addition, we propose a modification to establish the frequency of a subprocess. Due to the support for selections in the subprocesses, a subprocess can model more than one path. To ensure that an infrequent subprocess only models infrequent behavior, all its paths must be infrequent. For this reason, the frequency of a subprocess must be the highest of its path's frequencies.

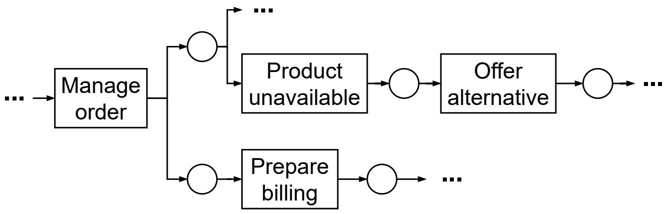
- As with the a-priori search of WoMine, WoMine-i retains all the infrequent subprocesses found through the expansion process. For this reason, the result of the main search contains subprocesses that model behavior already modeled by other subprocesses. To reduce the redundancy in the results, we propose a postprocessing stage to retain only the subprocesses which behavior is not supported by another subprocess. In this case, the subprocesses to retain are the minimal ones —to express with the smaller structure the more amount of behavior as possible. When the behavior of a subprocess is contained into another subprocess, a decision must be made to retain the smaller subprocess —if it supports all the paths modeled by the other subprocess— or the larger subprocess —if it models a path not supported by the smaller one.

Process model simplification

As we have already commented, many techniques have been proposed to simplify complex process models by removing the infrequent behavior. Nevertheless, this information can be useful in future phases of the analysis. The event abstraction [4, 28, 35] appears as a way to maintain in the model the presence of the abstracted behavior, while highlighting the visualization of the rest. In order to obtain an overall view of the process depicting the frequent behavior, while being aware of the existence and exact location of the infrequent behavior, we propose to perform an abstraction of the infrequent subprocesses by encapsulating them into artificial activities. We propose two alternatives to perform this abstraction: *i)* to identify the frequent subprocesses to retain, and abstract the remaining behavior, and *ii)* to identify the infrequent behavior to abstract, retaining the remaining behavior. As we will show throughout this dissertation, the discovery of infrequent subprocesses has a higher computational cost than the discovery of frequent subprocesses. This characteristic has a low impact in normal processes, where both alternatives could coexist. Nevertheless, it presents some disadvantages in complex processes, where this difference becomes more notorious. As our objective is to simplify complex process models, we have discarded the second alternative. Thus, our proposal consists of *i)* a search for the frequent subprocesses to retain, *ii)* an identification of the events related to these subprocesses, and *iii)* an abstraction of the remaining —infrequent— behavior.

Few techniques had been published using event abstraction in order to simplify complex process models, and none of them propose the abstraction of the infrequent behavior that is obfuscating the simplicity and precision of the process model. The third contribution of this Ph.D. Thesis is IBeA, an algorithm to simplify complex process models by abstracting the infrequent behavior. To perform this abstraction, we propose to first obtain the frequent subprocesses to retain by using WoMine. Then, we propose to identify the traces in which these subprocesses are executed, and to mark the events corresponding to their execution as the behavior to retain. Finally, we propose to encapsulate the unmarked events —i.e. those not corresponding to the frequent subprocesses— into artificial activities based on the relations present in the model. In this abstraction, the events corresponding to activities connected between them in the process model are abstracted together into one artificial activity. This abstraction gives the possibility to obtain, on the one hand, the simplified process model and, on the other hand, the corresponding simplified event log to discover a simplified process model. Besides, with the simplified event log, further process mining analyses can be performed.

One of the main contributions of IBeA is related to its proposal to abstract the infrequent behavior. IBeA takes advantage of the relations of the process model to connect the events between them. For example, consider the part of a process depicted in Figure 1.9, where the execution of the activities *Product unavailable* and *Offer alternative* are considered infrequent. After the execution of *Manage order*, a concurrent structure appears where *Prepare billing* can be registered interleaving with the other activities, as shown in Figure 1.9(b). *Product unavailable* and *Offer alternative* may not be adjacent —as shown in the second trace of the example— but, thanks to the relation in the process model (Figure 1.9(a)), we know that the events recording the execution of *Offer alternative* and *Product unavailable* are connected. IBeA exploits this information to perform an abstraction taking into account the structures



(a) Part of a process model.

Trace
... Manage order - Product unavailable - Offer alternative - Prepare billing ...
... Manage order - Product unavailable - Prepare billing - Offer alternative ...
... Manage order - Prepare billing - Product unavailable - Offer alternative ...

(b) Part of 3 traces corresponding the process model in Figure 1.9.

Figure 1.9: Partial example of a process model and an event log to show the advantage of using the process model relations to perform the abstraction.

of the model. Thus, the events of an infrequent path in the model —that can be registered interleaved with other events from a frequent subprocess— are abstracted together into one artificial activity. This characteristic ensures that the abstraction of an infrequent subprocess is performed by replacing all its events with one single artificial activity, even if these events are interleaved with events of a frequent subprocess.

We propose IBeA as a two-part algorithm. In the first stage, WoMine is used to extract the frequent subprocesses to retain, and identifies as *core behavior* the events corresponding to the execution of those subprocesses. In the second stage, an abstraction process is launched in which the *non-core behavior* is abstracted into artificial activities. We propose this second stage as an independent algorithm, called UBeA, being the fourth contribution of this Ph.D. Thesis. UBeA receives a process model, an event log, and the events corresponding to the core behavior, and abstracts the non-core behavior by replacing it with artificial activities taking into account its relations in the process model.

The potential of UBeA resides in its ability to abstract any behavior taking into account the relations of the process model. This gives the opportunity to perform the abstraction in order to hide the unimportant behavior, and observe the interaction between any set of subprocesses —identifying them as the core behavior. For example, consider a delivery company organized in different departments. The full process model might depict all the activities of the process, independently of which department is executing them. To observe the different departments, UBeA could be used by marking as core behavior the subprocesses related to some departments. In this way, the activities related to other departments would be abstracted, and the subprocesses of the desired departments would remain, along with their relations. In the same way, other analyses regarding the interaction between resources (organizational perspective) could be performed.

The potential of this abstraction is not only present in the organizational perspective. An abstraction of subprocesses could be also performed from the data perspective. For example, in order to analyze the less expensive parts of the process model, the events with cost values —or related to subprocesses with a cost value— under a predefined amount could be abstracted. In this way, the result of UBeA would be the process model depicting the subprocesses with a high cost, and how those subprocesses interact among them. The advantage is that the non-core behavior is not being removed, but abstracted into artificial activities. Thus, the process model depicts a general view of the important subprocesses —those with a high cost in this case—, but it also shows where the other subprocesses are being executed, encapsulated into artificial activities.

In summary, this Ph.D. Thesis proposes four algorithms: *i)* WoMine, an algorithm to discover the frequent subprocesses of a process model; *ii)* WoMine-i, an algorithm to discover the infrequent subprocesses of a process model; *iii)* IBeA, an algorithm to simplify process models by abstracting the infrequent behavior into artificial activities and; *iv)* UBeA, a generic algorithm to abstract non-core behavior in order to ease the analysis of the core behavior.

1.6 Research Contributions

The main contributions of this Ph.D. Thesis are as follows:

C1. WoMine - An algorithm for the extraction of frequent subprocesses.

We have developed an algorithm (WoMine) for the extraction of frequent subprocesses from a process model, measuring their frequency in the traces of the event log. WoMine is able to discover the frequent maximal subprocesses of a process model, supporting all types of structures —even n-length cycles—, and ensuring their frequency w.r.t. the traces of the event log. The contributions of WoMine are as follows:

- Discover frequent subprocesses contained in the process model with all types of structures (sequences, concurrency, choices, and loops).
- An a-priori search reducing drastically the search space.
- A novel way to measure the frequency of the subprocesses, ensuring that their execution is not disrupted by the execution of other activities.
- A post-processing step to retrieve only the maximal subprocesses, avoiding redundancy.

C2. WoMine-i - An algorithm for the extraction of infrequent subprocesses.

We have developed an algorithm (WoMine-i) for the extraction of infrequent subprocesses from a process model, measuring their frequency in the traces of the event log. WoMine-i is able to discover the infrequent minimal subprocesses of a process model, supporting all types of structures —even n-length cycles—, and ensuring their frequency w.r.t. the traces of the event log. The contributions of WoMine-i are as follows:

- Discover infrequent subprocesses contained in the process model with all types of structures (sequences, concurrency, choices, and loops).
- A novel way to measure the frequency of the subprocesses, ensuring that their execution is not disrupted by the execution of other activities.
- A post-processing step to retrieve only the minimal subprocesses, avoiding redundancy.

C3. UBeA - A generic algorithm for the abstraction of non-core behavior in process models.

We have developed an algorithm (UBeA) that, given *i)* an event log, *ii)* a process model, and *ii)* a set of events of the non-core behavior, abstracts in both the event log and the process model the non-core behavior by replacing it with artificial activities. UBeA presents a novel way of abstraction that encapsulates the non-core behavior into artificial activities maintaining its relations, providing a simplified view of the core behavior, but also depicting the non-core behavior.

C4. IBeA - An algorithm for the simplification of process models by abstracting the infrequent behavior.

We have developed an algorithm (IBeA) to simplify process models by abstracting the infrequent behavior which is harming the precision and simplicity of the process model. IBeA uses WoMine to extract the frequent subprocesses to maintain in the abstraction. Then, it identifies the events which do not correspond to the frequent subprocesses. Finally, by using UBeA, the infrequent behavior is abstracted into artificial activities,

C5. Software tools.

Tool - WoMine & WoMine-i Application. [1]

A web platform to discover and visualize the frequent and infrequent subprocesses discovered by WoMine and WoMine-i algorithms, respectively. This web application allows users to upload a process model, an event log, and to define a frequency threshold in order to discover either the frequent or infrequent subprocesses w.r.t. that threshold. The visualization of the subprocesses can be done by highlighting them in the full process model, or depicting only the subprocess. Furthermore, for an easy use, the application allows the user to filter the resulting subprocesses by the activities they contain.

Tool - InVerbis Analytics. [2]

InVerbis Analytics is a start-up that originated at CiTIUS, Universidade de Santiago de Compostela, offering tools to analyze business processes by using process mining and business process management techniques. Many of the algorithms and code produced during this Ph.D. Thesis have been integrated into the InVerbis application.

Tool - KTPM - A Library for Process Mining in Kotlin.

During the last years of this Ph.D. Thesis, as a collaboration with Victor Gallego-Fontenla, the implementation of the developed algorithms has been designed as part of a framework for process mining in Kotlin. We have implemented functionalities to read/write event logs and process models, discovery algorithms, conformance checking techniques, concept drift algorithms, and other utilities to transform event logs and extract different statistics from them. The framework is still under development.

[1] `https://tec.citius.usc.es/graphmining/`
[2] `https://web.inverbisanalytics.com/`

1.7 Publications

The contributions of this Ph.D. Thesis are included in the following publications:

Journal Papers

Inf. Sci. D. Chapela-Campa, M. Mucientes, and M. Lama. Mining frequent patterns in process models. *Information Sciences*, 472:235–257, 2019. (DOI 10.1016/j.ins.2018.09.011).

- Impact Factor (JCR 2019): 5.910. Category: COMPUTER SCIENCE, INFORMATION SYSTEMS. Rank: 9/156 (Q1).

Future Gener. Comput. Syst. D. Chapela-Campa, M. Mucientes, and M. Lama. Understanding complex process models by abstracting infrequent behavior. *Future Generation Computer Systems*, 113:428–440, 2020. (DOI 10.1016/j.future.2020.07.030).

- Impact Factor (JCR 2020): 7.187. Category: COMPUTER SCIENCE, THEORY & METHODS. Rank: 7/110 (Q1).

International Conferences

BPM D. Chapela-Campa, M. Mucientes, and M. Lama. Discovering Infrequent Behavioral Patterns in Process Models. In *15th International Conference on Business Process Management (BPM 2017)*, volume 10445 of *Lecture Notes in Computer Science*, pages 324–340, Springer, 2017. (DOI 10.1007/978-3-319-65000-5_19).

- Conference Ranking: GGS Class 2 (GGS Class 2018), GGS Rating A (GGS Rating 2018), CORE A (CORE 2017).

ICSOC D. Chapela-Campa, M. Mucientes, and M. Lama. Simplification of Complex Process Models by Abstracting Infrequent Behaviour. In *17th International Conference on Service-Oriented Computing (ICSOC 2019)*, volume 11895 of *Lecture Notes in Computer Science*, pages 415–430, Springer, 2019. (DOI 10.1007/978-3-030-33702-5_32).

- Conference Ranking: GGS Class 2 (GGS Class 2018), GGS Rating A- (GGS Rating 2018), CORE A (CORE 2020).

ATAED B. Vazquez-Barreiros, D. Chapela, M. Mucientes, and M. Lama. Process Mining in IT Service Management: A Case Study. In *2016 International Workshop on Algorithms & Theories for the Analysis of Event Data, (ATAED 2016)*, volume 1592 of *CEUR Workshop Proceedings*, pages 16–30, 2016.

National Conferences

JCIS D. Chapela-Campa, M. Mucientes, and M. Lama. Towards the Extraction of Frequent Patterns in Complex Process Models. *Jornadas de Ciencia e Ingeniería de Servicios, JCIS*, 2017.

(HANDLE 11705/JCIS/2017/006).

JCIS D. Chapela-Campa, M. Mucientes, and M. Lama. Discovering Infrequent Behavioral Patterns in Process Models (Summary). *Jornadas de Ciencia e Ingeniería de Servicios, JCIS*, 2018.

(HANDLE 11705/JCIS/2018/006).

JCIS D. Chapela-Campa, M. Mucientes, and M. Lama. Pattern-based Simplification of Process Models. *Jornadas de Ciencia e Ingeniería de Servicios, JCIS*, 2019.

(HANDLE 11705/JCIS/2019/019).

1.8 Thesis Outline

This Ph.D. Thesis is structured into five chapters. Chapters 2, 3, and 4 detail each of the proposed techniques and their experimentation results. Finally, Chapter 5 draws the conclusions and future work of this research. More specifically, the dissertation structure is as follows:

- Chapter 2 introduces the topic of frequent behavior extraction and the contributions of this Ph.D. Thesis to this field. This chapter describes WoMine, an algorithm to discover maximal frequent subprocesses from a process model, measuring their frequency in the instances of the event log.

- Chapter 3 presents the contributions to the search for infrequent behavior from processes. This chapter describes WoMine-i, an algorithm to detect minimal infrequent subprocesses from a process model, measuring their frequency with the instances of the event log.

- Chapter 4 addresses the task of complex process model simplification. This chapter describes two algorithms for behavioral abstraction in process models: UBeA and IBeA. UBeA is an algorithm to abstract the non-core behavior —specified by the user— of a process model into artificial activities using the relations among the activities. On the other hand, IBeA is a specific implementation of UBeA to simplify process models by abstracting infrequent behavior, using WoMine to detect the core behavior —considering the infrequent behavior as non-core—, allowing to produce a simpler process model while maintaining a good trade-off between fitness and precision.

- Chapter 5 outlines the conclusions of this Ph.D. Thesis and sketches future work directions.

Chapter 2

Mining frequent patterns in process models

"Essentially, all models are wrong, but some are useful."

— George E. P. Box

As we have already discussed in Chapter 1, the extraction of frequent behavior is crucial in order to analyze and simplify complex processes where the discovered process model presents low simplicity and precision values. Many techniques have been developed focusing on the discovery of frequent subprocesses —or frequent patterns. Only a few of these techniques support structures like concurrency and loops. Nevertheless, all of them measure the frequency of the subprocesses by taking into account only the activities forming them. As we have shown, this characteristic may lead to misconclusions and misinterpretations in some cases.

To cope with these shortcomings, in this chapter we introduce WoMine, an algorithm to discover maximal frequent subprocesses from a process model, measuring their frequency in the instances of the event log. WoMine discovers frequent subprocesses with all type of structures —even n-length cycles, very common structures in real processes. The algorithm performs an a-priori search building the subprocesses with the relations in the process model, reducing in this way the search space. Furthermore, WoMine measures the frequency of each subprocess by taking into account the relations of the process model, ensuring a precise frequency measurement. The algorithm has been tested with 20 synthetic process models ranging from 20 to 30 unique activities, and containing sequences, concurrency, selections, and loops. Experiments have been also run with 12 real complex event logs of the Business Process Intelligence Challenges.

This chapter includes a full copy of the following journal paper that describes in detail the proposed approach:

> D. Chapela-Campa[†], M. Mucientes[†], and M. Lama[†]. Mining frequent patterns in process models. *Information Sciences*, 472:235–257, 2019.
> (DOI 10.1016/j.ins.2018.09.011)

Publishing rights

According to Elsevier copyright terms, the authors have the right to include the accepted version of Elsevier-copyrighted articles on their thesis or dissertation (provided this is not published commercially).

[†]Centro Singular de Investigación en Tecnoloxías Intelixentes (CiTIUS), Universidade de Santiago de Compostela. Santiago de Compostela, Spain.

Chapter 3

Discovering Infrequent Behavioral Patterns in Process Models

“The invisible and the non-existent look very much alike."
— Delos Banning McKown

In Chapter 2 we presented an algorithm for the discovery of frequent subprocesses from process models. As we have stated in Chapter 1, the detection of infrequent subprocesses can be combined with the detection of frequent subprocesses in order to optimize the simplification. Furthermore, the analysis of infrequent behavior can be useful on its own. There are scenarios where an infrequent subprocess can hint at erroneous behavior which must be examined. Many techniques have been developed focusing in the detection of infrequent (or outlier) traces and events, but none of them search for infrequent subprocesses.

In order to discover the infrequent subprocesses of a process, in this chapter we introduce WoMine-i, an algorithm to detect minimal infrequent subprocesses from a process model, measuring their frequency with the instances of the event log. The main novelty of our approach is that it can detect infrequent subprocesses with all types of structures —sequences, selections, concurrency, and loops. The ability to work with these structures prevents WoMine-i from interpreting the traces as sequences of events. Furthermore, the extracted information allows to focus on infrequent subprocesses, and not to analyze infrequent full traces. The algorithm has been qualitatively compared using various synthetic process models with all related techniques. Experiments have also been conducted with real event logs of two Business Process Intelligence Challenges, BPIC 2012 and BPIC 2013.

This chapter includes a full copy of the following conference paper that describes in detail the proposed approach:

> D. Chapela-Campa[†], M. Mucientes[†], and M. Lama[†]. Discovering Infrequent Behavioral Patterns in Process Models. In *15th International Conference on Business Process Management (BPM 2017)*, volume 10445 of *Lecture Notes in Computer Science*, pages 324–340, Springer, 2017.
> (DOI 10.1007/978-3-319-65000-5_19).

Publishing rights

According to Springer copyright terms, the authors have the right to include the accepted version of their Springer-copyrighted article's Version of Record, in whole or in part, in their own thesis. Additionally, they may reproduce and make available their thesis, including Springer Nature content, as required by their awarding academic institution.

[†]Centro Singular de Investigación en Tecnoloxías Intelixentes (CiTIUS), Universidade de Santiago de Compostela. Santiago de Compostela, Spain.

Chapter 4

Understanding complex process models by abstracting infrequent behavior

"The height of sophistication is simplicity."

— Clare Boothe Luce

The main objective of this Ph.D. Thesis is the simplification of complex process models. As we have introduced in Chapter 1, with more information and behavior related to the processes being recorded, the apparition of complex processes and complex process models —with hundreds of edges and activities— has become more common. The complexity of these discovered process models —characterized by low simplicity and precision— hinder the detection of deviations during conformance, and the posterior enhancement. Furthermore, most stages of BPM depending on the process model, such as the analysis, redesign, and implementation, are also hindered. Many techniques have been developed in order to simplify complex processes during past years. Nevertheless, most of them focus on the removal of the infrequent behavior, either full traces, single events, or activities. As it has been discussed, the removal of behavior from the process model can lead to misconclusions in some contexts.

To cope with these shortcomings, in this chapter we present two algorithms for behavioral abstraction in process models: UBeA and IBeA. UBeA is an algorithm to abstract the non-core behavior of a process into artificial activities using the relations between the activities, hence, taking into account structures such as concurrency, selections, or loops. The main novelty of UBeA is that it generates an abstracted version of the process describing the core behavior while being aware of the existence and exact location of the non-core behavior. Furthermore, UBeA allows the user to specify the behavior to maintain, i.e., the core behavior, making it very versatile. On the other hand, IBeA is a specific implementation of UBeA to simplify process models by abstracting infrequent behavior, using WoMine to detect the core behavior —considering the infrequent behavior as non-core— allowing to produce a simpler process model while maintaining a trade-off between fitness and precision. IBeA has been validated with a set of 11 complex and real event logs, 10 of them from the Business Process Intelligence Challenges (BPIC), and one from the health domain.

This chapter includes a full copy of the following journal paper that describes in detail the proposed approach:

> D. Chapela-Campa[†], M. Mucientes[†], and M. Lama[†]. Understanding complex process models by abstracting infrequent behavior. *Future Generation Computer Systems*, 113:428–440, 2020.
> (DOI 10.1016/j.future.2020.07.030)

Publishing rights

According to Elsevier copyright terms, the authors have the right to include the accepted version of Elsevier-copyrighted articles on their thesis or dissertation (provided this is not published commercially).

[†]Centro Singular de Investigación en Tecnoloxías Intelixentes (CiTIUS), Universidade de Santiago de Compostela. Santiago de Compostela, Spain.

CHAPTER 5

CONCLUSIONS

"There is no real ending. It's just the place where you stop the story."

— Frank Herbert

In this Ph.D. Thesis, we have addressed the problem of complex process model simplification through the use of frequent and infrequent behavior-based algorithms. In past years, with the explosion of process-related data gathering, the apparition of complex processes and complex process models —with hundreds of edges and activities— has become more common. We have introduced the problems these types of process models can cause to the analyses performed in most of the stages of Business Process Management and Process Mining. Therefore, in past years there has been increasing interest in the simplification of complex process models.

We have focused on the simplification of complex process models by abstracting the infrequent behavior. For this purpose, we first developed WoMine, an algorithm for the discovery of maximal frequent subprocesses from a process model. We have designed an a-priori search —similar to *w*-find [20]— to take advantage of the relations between the activities in the process model and reduce the search space. We have proved qualitatively that, in some contexts, existent techniques fail in the measurement of a subprocess' frequency due to only considering its activities to check if the subprocess is executed. As it has been shown, this might be harmful in some cases by leading to wrong conclusions and misinterpretations. To cope with this shortcoming, we have proposed a method to measure the frequency of a subprocess by taking into account all the activities of the process, and the relations among them supported by the process model. Finally, we have also proposed a procedure to remove the redundancy in the discovered frequent subprocesses by retaining the maximal ones. We have performed an experimentation to qualitatively prove the superiority of WoMine over the state of the art techniques, to show the ability of WoMine to extract subprocesses with all types of structures —even n-length cycles— in both synthetic and real process models, and to analyze its potential by depicting some of the obtained results.

In addition, we have also developed WoMine-i, an algorithm for the discovery of infrequent subprocesses from a process model. WoMine-i takes advantage of the a-priori search of WoMine, and adapts it by changing the pruning strategy in order to find infrequent subprocesses. To ensure a strict measurement of the subprocesses' frequency, we have used the method proposed in WoMine. We have also developed a new method to reduce the redundancy in the set of infrequent subprocesses. There is no previous work regarding the discovery of infrequent subprocesses. For this reason, we have compared WoMine-i with techniques focused on the detection of infrequent behavior —in the form of complete traces or activities—, and with frequent subprocess discovery algorithms —modified to search for infrequent subprocesses. Furthermore, we have shown the ability of WoMine-i to discover subprocesses with all types of structures —even n-length cycles— with a set of real process models, and we have analyzed its potential by depicting some of the obtained results.

Finally, to tackle the problem of complex process model simplification, we have developed IBeA. IBeA is an algorithm to simplify complex process models by abstracting the infrequent behavior into artificial activities. In this way, IBeA maintains the frequent subprocesses in the process model while abstracting the infrequent behavior that is obfuscating the visualization of the process model. For this, the events corresponding to the execution of frequent subprocesses are identified with the use of WoMine. Then, the other events are abstracted by encapsulating them into artificial activities based on the relations present in the model —the events of an infrequent subprocess connected between them are replaced together by one single artificial activity. IBeA produces a simplified event log and a simplified process model, both with the infrequent behavior abstracted into artificial activities. The simplified event log gives the opportunity to perform different process mining analyses, and to rediscover the simplified process model with a discovery algorithm. We have used 11 real-life event logs to compare the simplification performance of IBeA —both by obtaining the simplified process model and by discovering a process model from the simplified event log— with the state of the art techniques. Results prove that, in complex event logs with high variability in the recorded behavior, the simplification of IBeA obtains a simpler process model with a better trade-off between fitness and precision. Furthermore, we have shown the potential of IBeA simplification by analyzing some of the simplified process models.

In addition, we have designed IBeA as a two-part algorithm: *i)* the search for the frequent subprocesses and the identification of its events; and *ii)* the abstraction of the unidentified behavior. We have developed this second stage as a generic algorithm to abstract non-core behavior, called UBeA. UBeA receives the event log, the process model, and the events corresponding to the behavior to retain —i.e. the core behavior—, and abstracts the rest by encapsulating it into artificial activities based on their relations in the process model. UBeA is useful to analyze the process retaining any type of subprocesses —frequent subprocesses, subprocesses related to a single department, subprocesses with a high cost, etc.— by reducing the other behavior into artificial activities. In this way, the analysis can focus on the desired subprocesses, while being aware of the relations between them and the existence and exact location of the other behavior, which is abstracted into artificial activities.

Future work

The research accomplished in this Ph.D. Thesis has opened some directions that might be interesting to explore in the future:

- *Explore the scalability of infrequent behavior search.* As it has been shown, the search space in the infrequent subprocess search grows too fast to be scalable in complex processes. More work could be done regarding the pruning strategy of WoMine-i in order to refine it and improve the scalability of the technique. One of these improvements could be to design the search without the support for selections, avoiding the possibility of having more than one path in a subprocess. In this way, the pruning strategy could consist only of pruning a subprocess at the moment it becomes infrequent —as there is no need to continue expanding it in order to model other paths. Nevertheless, to reduce the redundancy in the resultant subprocesses, a postprocessing phase to merge those subprocesses with common parts would be necessary.

- *Improve the naming of the abstracted activities.* Currently, IBeA names the artificial activities with a label to identify that they are artificial activities encapsulating infrequent behavior. Techniques to produce a name based on the encapsulated behavior could be proposed, for example with the use of semantic annotation techniques [36]. In this way, a more descriptive name could be automatically assigned to the abstractions, helping further analyses.

- *Increase the information of the abstracted activities.* IBeA abstracts the infrequent behavior into artificial activities. The result is a process model and an event log with these artificial activities encapsulating the abstracted behavior —i.e. the replaced events. No information regarding the abstracted behavior is given by the algorithm. Nevertheless, the direct inspection of the abstracted subtraces —i.e. the events that have been replaced— could be easily performed. Furthermore, more refined analyses could be proposed by applying discovery techniques to this infrequent behavior. For example, a search for frequent subprocesses inside the infrequent behavior, the discovery of declarative rules that define the characteristics of this abstracted behavior [18, 24, 40], etc.

- *Exploit the potential of UBeA.* As it has been shown, the abstraction of behavior itself has been designed as an independent algorithm, called UBeA. UBeA receives the event log, the process model, and the set of events to retain —i.e. the frequent behavior in the case of IBeA—, and abstracts the remaining behavior into artificial activities by taking into account the relations present in the process model. This abstraction can be used for many purposes. For example, to focus the analysis on a specific type of subprocesses —e.g. the subprocesses associated to a high cost— and abstract from the other behavior. It is interesting to explore different applications of UBeA's abstraction to exploit its potential.

Bibliography

[1] Rakesh Agrawal and Ramakrishnan Srikant. Mining sequential patterns. In *11th International Conference on Data Engineering (ICDE 1995)*, pages 3–14. IEEE Computer Society, 1995.

[2] Abel Armas-Cervantes, N. R. T. P. van Beest, Marcello La Rosa, Marlon Dumas, and Luciano García-Bañuelos. Interactive and incremental business process model repair. In *On the Move to Meaningful Internet Systems. OTM 2017 Conferences - 25th International Conference on Cooperative Information Systems (CoopIS 2017), Part I*, volume 10573 of *LNCS*, pages 53–74. Springer, 2017.

[3] Adriano Augusto, Raffaele Conforti, Marlon Dumas, Marcello La Rosa, and Artem Polyvyanyy. Split miner: automated discovery of accurate and simple business process models from event logs. *Knowl. Inf. Syst.*, 59(2):251–284, 2019.

[4] R. P. Jagadeesh Chandra Bose and Wil M. P. van der Aalst. Abstractions in process mining: A taxonomy of patterns. In *7th International Conference on Business Process Management (BPM 2009)*, volume 5701 of *LNCS*, pages 159–175. Springer, 2009.

[5] Dang Bach Bui, Fedja Hadzic, and Vidyasagar M. Potdar. A framework for application of tree-structured data mining to process log analysis. In *13th International Conference on Intelligent Data Engineering and Automated Learning (IDEAL 2012)*, volume 7435 of *LNCS*, pages 423–434. Springer, 2012.

[6] Joos C. A. M. Buijs, Boudewijn F. van Dongen, and Wil M. P. van der Aalst. On the role of fitness, precision, generalization and simplicity in process discovery. In *On the Move to Meaningful Internet Systems. OTM 2012 Conferences - 20th International*

Conference on Cooperative Information Systems (CoopIS 2012), Part I, volume 7565 of *LNCS*, pages 305–322. Springer, 2012.

[7] Josep Carmona, Boudewijn F. van Dongen, Andreas Solti, and Matthias Weidlich. *Conformance Checking - Relating Processes and Models*. Springer, 2018.

[8] Michelangelo Ceci, Pasqua Fabiana Lanotte, Fabio Fumarola, Dario Pietro Cavallo, and Donato Malerba. Completion time and next activity prediction of processes using sequential pattern mining. In *17th International Conference on Discovery Science (DS 2014)*, volume 8777 of *LNCS*, pages 49–61. Springer, 2014.

[9] Raffaele Conforti, Marcello La Rosa, and Arthur H. M. ter Hofstede. Filtering out infrequent behavior from business process event logs. *IEEE Trans. Knowl. Data Eng.*, 29(2):300–314, 2017.

[10] Benjamin Dalmas, Niek Tax, and Sylvie Norre. Heuristic mining approaches for high-utility local process models. *Trans. Petri Nets Other Model. Concurr.*, 13:27–51, 2018.

[11] Fábio de Lima Bezerra and Jacques Wainer. Algorithms for anomaly detection of traces in logs of process aware information systems. *Inf. Syst.*, 38(1):33–44, 2013.

[12] Javier de San Pedro, Josep Carmona, and Jordi Cortadella. Log-based simplification of process models. In *13th International Conference on Business Process Management (BPM 2015)*, volume 9253 of *LNCS*, pages 457–474. Springer, 2015.

[13] Jörg Desel and Wolfgang Reisig. Place or transition petri nets. In *Lectures on Petri Nets I: Basic Models, Advances in Petri Nets*, volume 1491 of *LNCS*, pages 122–173. Springer, 1996.

[14] Prabhakar M. Dixit, H. M. W. Verbeek, Joos C. A. M. Buijs, and Wil M. P. van der Aalst. Interactive data-driven process model construction. In *37th International Conference on Conceptual Modeling (ER 2018)*, volume 11157 of *LNCS*, pages 251–265. Springer, 2018.

[15] Marlon Dumas, Marcello La Rosa, Jan Mendling, and Hajo A. Reijers. *Fundamentals of Business Process Management, Second Edition*. Springer, 2018.

[16] Dirk Fahland and Wil M. P. van der Aalst. Simplifying discovered process models in a controlled manner. *Inf. Syst.*, 38(4):585–605, 2013.

[17] Stefano Ferilli and Sergio Angelastro. Activity prediction in process mining using the woman framework. *J. Intell. Inf. Syst.*, 53(1):93–112, 2019.

[18] Giuseppe De Giacomo, Marlon Dumas, Fabrizio Maria Maggi, and Marco Montali. Declarative process modeling in BPMN. In *27th International Conference on Advanced Information Systems Engineering (CAiSE 2015)*, volume 9097 of *LNCS*, pages 84–100. Springer, 2015.

[19] Gianluigi Greco, Antonella Guzzo, Francesco Lupia, and Luigi Pontieri. Process discovery under precedence constraints. *ACM Trans. Knowl. Discov. Data*, 9(4):32:1–32:39, 2015.

[20] Gianluigi Greco, Antonella Guzzo, Giuseppe Manco, Luigi Pontieri, and Domenico Saccà. Mining constrained graphs: The case of workflow systems. In *2004 European Workshop on Inductive Databases and Constraint Based Mining, Revised Selected Papers*, volume 3848 of *LNCS*, pages 155–171. Springer, 2004.

[21] Wai Lam Jonathan Lee, H. M. W. Verbeek, Jorge Munoz-Gama, Wil M. P. van der Aalst, and Marcos Sepúlveda. Recomposing conformance: Closing the circle on decomposed alignment-based conformance checking in process mining. *Inf. Sci.*, 466:55–91, 2018.

[22] Maikel Leemans and Wil M. P. van der Aalst. Discovery of frequent episodes in event logs. In *4th International Symposium on Data-Driven Process Discovery and Analysis (SIMPDA 2014), Revised Selected Papers*, volume 237 of *LNBIP*, pages 1–31. Springer, 2014.

[23] Sander J. J. Leemans, Dirk Fahland, and Wil M. P. van der Aalst. Discovering block-structured process models from event logs containing infrequent behaviour. In *11th International Workshops on Business Process Management (BPM 2013), Revised Papers*, volume 171 of *LNBIP*, pages 66–78. Springer, 2013.

[24] Volodymyr Leno, Marlon Dumas, Fabrizio Maria Maggi, Marcello La Rosa, and Artem Polyvyanyy. Automated discovery of declarative process models with correlated data conditions. *Inf. Syst.*, 89:101482, 2020.

[25] David Lo, Hong Cheng, Jiawei Han, Siau-Cheng Khoo, and Chengnian Sun. Classification of software behaviors for failure detection: a discriminative pattern mining

approach. In *15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2009)*, pages 557–566. ACM, 2009.

[26] Xixi Lu, Dirk Fahland, Frank J. H. M. van den Biggelaar, and Wil M. P. van der Aalst. Detecting deviating behaviors without models. In *13th International Workshops on Business Process Management (BPM 2015), Revised Papers*, volume 256 of *LNBIP*, pages 126–139. Springer, 2015.

[27] Felix Mannhardt. Multi-perspective process mining. In *Proceedings of the Dissertation Award, Demonstration, and Industrial Track at 16th International Conference on Business Process Management (BPM 2018)*, volume 2196 of *CEUR Workshop Proceedings*, pages 41–45. CEUR-WS.org, 2018.

[28] Felix Mannhardt, Massimiliano de Leoni, Hajo A. Reijers, Wil M. P. van der Aalst, and Pieter J. Toussaint. From low-level events to activities - A pattern-based approach. In *14th International Conference on Business Process Management (BPM 2016)*, volume 9850 of *LNCS*, pages 125–141. Springer, 2016.

[29] Felix Mannhardt and Niek Tax. Unsupervised event abstraction using pattern abstraction and local process models. In *18th International Working Conference on Business Process Modeling, Development and Support (BPMDS 2017)*, volume 1859 of *CEUR Workshop Proceedings*, pages 55–63. CEUR-WS.org, 2017.

[30] Heikki Mannila, Hannu Toivonen, and A. Inkeri Verkamo. Discovery of frequent episodes in event sequences. *Data Min. Knowl. Discov.*, 1(3):259–289, 1997.

[31] Jorge Munoz-Gama. *Conformance Checking and Diagnosis in Process Mining - Comparing Observed and Modeled Processes*, volume 270 of *LNBIP*. Springer, 2016.

[32] Gerhard Münz, Sa Li, and Georg Carle. Traffic anomaly detection using k-means clustering. In *GI/ITG Workshop MMBnet 2007*, pages 13–14, 2007.

[33] Jian Pei, Jiawei Han, Behzad Mortazavi-Asl, Helen Pinto, Qiming Chen, Umeshwar Dayal, and Meichun Hsu. Prefixspan: Mining sequential patterns by prefix-projected growth. In *17th International Conference on Data Engineering (ICDE 2001)*, pages 215–224. IEEE Computer Society, 2001.

[34] James L. Peterson. Petri nets. *ACM Comput. Surv.*, 9(3):223–252, 1977.

[35] Artem Polyvyanyy, Sergey Smirnov, and Mathias Weske. Process model abstraction: A slider approach. In *12th International IEEE Enterprise Distributed Object Computing Conference (ECOC 2008)*, pages 325–331. IEEE Computer Society, 2008.

[36] Efrén Rama-Maneiro, Juan Carlos Vidal, and Manuel Lama. Collective disambiguation in entity linking based on topic coherence in semantic graphs. *Knowl. Based Syst.*, 199:105967, 2020.

[37] Manuel Caeiro Rodríguez, Martín Llamas Nistal, Fernando A. Mikic-Fonte, Manuel Lama Penín, and Manuel Mucientes Molina. Exploring the application of process mining to support self-regulated learning: An initial analysis with video lectures. In *2018 IEEE Global Engineering Education Conference (EDUCON 2018)*, pages 1766–1774. IEEE, 2018.

[38] Nick Russell, Wil M. P. van der Aalst, and Arthur H. M. ter Hofstede. *Workflow Patterns: The Definitive Guide*. MIT Press, 2016.

[39] Mohammadreza Fani Sani, Sebastiaan J. van Zelst, and Wil M. P. van der Aalst. Improving process discovery results by filtering outliers using conditional behavioural probabilities. In *15th International Workshops on Business Process Management (BPM 2017), Revised Papers*, volume 308 of *LNBIP*, pages 216–229. Springer, 2017.

[40] Stefan Schönig, Claudio Di Ciccio, Fabrizio Maria Maggi, and Jan Mendling. Discovery of multi-perspective declarative process models. In *14th International Conference on Service-Oriented Computing (ICSOC 2016)*, volume 9936 of *LNCS*, pages 87–103. Springer, 2016.

[41] Daniel Schuster, Sebastiaan J. van Zelst, and Wil M. P. van der Aalst. Incremental discovery of hierarchical process models. In *14th International Conference on Research Challenges in Information Science (RCIS 2020)*, volume 385 of *LNBIP*, pages 417–433. Springer, 2020.

[42] Niek Tax, Natalia Sidorova, Reinder Haakma, and Wil M. P. van der Aalst. Mining local process models. *J. Innov. Digit. Ecosyst.*, 3(2):183–196, 2016.

[43] Niek Tax, Natalia Sidorova, Reinder Haakma, and Wil M. P. van der Aalst. Mining local process models with constraints efficiently: Applications to the analysis of smart home

data. In *14th International Conference on Intelligent Environments (IE 2018), Roma, Italy, June 25-28, 2018*, pages 56–63. IEEE, 2018.

[44] Niek Tax, Natalia Sidorova, and Wil M. P. van der Aalst. Discovering more precise process models from event logs by filtering out chaotic activities. *J. Intell. Inf. Syst.*, 52(1):107–139, 2019.

[45] Wil M. P. van der Aalst. *Process Mining - Data Science in Action, Second Edition*. Springer, 2016.

[46] Wil M. P. van der Aalst, M. H. Schonenberg, and Minseok Song. Time prediction based on process mining. *Inf. Syst.*, 36(2):450–475, 2011.

[47] Boudewijn van Dongen. Bpi challenge 2020, Mar 2020.

[48] Borja Vázquez-Barreiros. *Mining complete, precise and simple process models*. PhD thesis, Universidade de Santiago de Compostela, 2016.

[49] Borja Vázquez-Barreiros, Manuel Mucientes, and Manuel Lama. Prodigen: Mining complete, precise and minimal structure process models with a genetic algorithm. *Inf. Sci.*, 294:315–333, 2015.

[50] Wan-Shiou Yang and San-Yih Hwang. A process-mining framework for the detection of healthcare fraud and abuse. *Expert Syst. Appl.*, 31(1):56–68, 2006.

List of Figures

List of Tables