UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

CENTRO INTERNACIONAL DE ESTUDOS
DE DOUTORAMENTO E AVANZADOS
DA USC (CIEDUS)

TESIS DE DOCTORADO

REMAINING TIME ESTIMATION IN BUSINESS PROCESSES USING TRACES' STRUCTURAL INFORMATION

Presentada por:

Ahmad Abdel Karim Ali Aburomman

Dirigida por:

Alberto J. Bugarín Diz
Manuel Lama Penín

ESCUELA DE DOCTORADO INTERNACIONAL

PROGRAMA DE DOCTORADO EN INVESTIGACIÓN EN TECNOLOXÍAS DA INFORMACIÓN

Centro Singular de Investigación en Tecnoloxías Intelixentes (CiTIUS)

SANTIAGO DE COMPOSTELA
31 de enero de 2020

USC
UNIVERSIDADE
DE SANTIAGO
DE COMPOSTELA

CENTRO INTERNACIONAL DE ESTUDOS
DE DOUTORAMENTO E AVANZADOS
DA USC (CIEDUS)

DECLARACIÓN DEL AUTOR DE LA TESIS
REMAINING TIME ESTIMATION IN BUSINESS PROCESSES USING TRACES' STRUCTURAL INFORMATION

Ahmad Abdel Karim Ali Aburomman

Presento mi tesis, siguiendo el procedimiento adecuado al Reglamento, y declaro que:

1. *La tesis abarca los resultados de la elaboración de mi trabajo.*
2. *En su caso, en la tesis se hace referencia a las colaboraciones que tuvo este trabajo.*
3. *La tesis es la versión definitiva presentada para su defensa y coincide con la versión enviada en formato electrónico.*
4. *Confirmo que la tesis no incurre en ningún tipo de plagio de otros autores ni de trabajos presentados por mí para la obtención de otros títulos.*

En Santiago de Compostela, 31 de enero de 2020

Fdo. Ahmad Abdel Karim Ali Aburomman

USC UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

CENTRO INTERNACIONAL DE ESTUDOS
DE DOUTORAMENTO E AVANZADOS
DA USC (CIEDUS)

AUTORIZACIÓN DEL DIRECTOR/TUTOR DE LA TESIS

REMAINING TIME ESTIMATION IN BUSINESS PROCESSES USING TRACES' STRUCTURAL INFORMATION

Dr. Alberto J. Bugarín Diz, Catedrático de Universidad del Área de Ciencia de la Computación e Inteligencia Artificial de la Universidade de Santiago de Compostela

Dr. Manuel Lama Penín, Profesor Titular de Universidad del Área de Ciencia de la Computación e Inteligencia Artificial de la Universidade de Santiago de Compostela

INFORMAN:

Que la presente tesis, corresponde con el trabajo realizado por ***Ahmad Abdel Karim Ali Aburomman*** *bajo nuestra dirección, y autorizamos su presentación, considerando que reúne los requisitos exigidos en el Reglamento de Estudios de Doctorado de la USC, y que como directores de ésta no incurre en las causas de abstención establecidas en Ley 40/2015.*

En Santiago de Compostela, 31 de enero de 2020

Fdo. Alberto J. Bugarín Diz
Director de la tesis

Fdo. Manuel Lama Penín
Director de la tesis

Dedication

To my Mother

A strong and gentle soul who taught me to believe in hard work and that so much could be done with little.

To my Father

For earning an honest living for us and for supporting and encouraging me to believe in myself.

To my beloved wife who has a special place in my heart…

Acknowledgments

In the name of Allah, the Most Merciful and Beneficent First and Foremost, praise is to ALLAH, the Almighty, the greatest of all, on whom ultimately, we depend for sustenance and guidance. I want to thank Almighty Allah for giving me the opportunity, determination, and strength to do my research.

My written words below could not be able to express my sincere gratitude from my deep heart which I owe to many people who have been of great help to me. But in many various manners, I will try to express my gratefulness to; all who help me, support me and encourage me during this fascinating experience. I would say sorry for anyone I forget if my memory betrayed me!

First, I would like to thank, as sincerely as warmly, my two supervisors. My supervisor Dr. Alberto Bugarín Diz I wish to express to you about my best gratitude first for your trust, I am grateful to you because you have encouraged me since the beginning of this thesis project. In short, I was lucky to work with you!!!

This work was supervised as well by Dr. Manuel Lama, to him, I send my special thanks and warm gratitude. It was such a delight to work with you Manuel, I am thankful for having welcomed me within your team. Thank you for your encouragement, patience and for the time you gave me while you were very busy.

I thank you both for the time you gave me to discuss the process of my work, during all my stay in CiTIUS. I am grateful to you for integrating me in the project: Ministerio de Ciencia, Innovación y Universidades (Refs. TIN2014-56633-C3-1-R and TIN2017-84796-C2-1-R,) and Consellería de Educación, Universidades e Formación Profesional da Xunta de Galicia (Refs. ED431C 2018/29, “acreditación 2016-2019, ED431G/08” and ED431G2019/04), all of them also supported by the Fondo Europeo de Desarrollo Regional del programa ERDF/FEDER. which was one of capital help for me, mainly in the aim of the important and condensate pa-

pers that we started to achieve together. ... Las palabras no son suficientes para expresar mi agradecimiento, ¡¡¡Muchas Gracias!!!

I would like to thank all my colleagues at the Centro Singular de Investigación en Tecnoloxías Intelixentes (CiTIUS) at the Universidade de Santiago de Compostela. Especially, Alberto Suarez Garea, Brais Bosquet Mera, Mauro Fernández Sanjurjo, Rodrigo Martínez Castaño, and all of the support staff (Secretary and technical employees). A special thank to all my Jordanian colleagues for all the support and grasp, Mohammad Alzubi, Anas Alrahamneh, Tareq Alzubi, Ahmad Abu Shattal, Firas Awaisheh, Zaid Lami, Saadi Alabadi, Sattam Almatarneh, Ali Alossofi, Ameed Almomani, Ala'a Alnuimat, Bilal Altawarah, Hatem Alameryeen, Ikremah Alqudah, Khalid Alkharabsheh, Ma'un Altawarah, Mutaz Almobydeen, Shahed Almobydeen, Taha Almarayeh, Abdulatif Almarayeh, Zakria Altarawneh, Mohammad Aladwan, Marwan Alajoury, Saqer Abu Shatal, Ali Alwardat, Mohammad Alda'ajeh, and Haitham Ma'arof. To my various and dear friends during my stay in Santiago de Compostela, Dr. Khalid Abu Amira, Dr. Sami Ashour, Dr. Hisham Atiyat, El Sayed Sadek Tohffah, Walid Sima.

I would like to express my deep and special gratitude to my small family; My dear parents Nuha and Abdelkarim, who have always been beside me in Weal and Woe! You are strong pillars who support me morally and financially, you follow me in my choices and my various adventures around the world. Thank you, mother and father, especially for your patience!!

My sisters Sanaa, Oula, Salam, and Dima for your support and patience!! My dear brothers Mohamad, Ghaith, Aiman, Housam, and Assem for supporting and continuous encouragement!

I will not forget to thank all of my aunts, uncles, and cousins for their support. Also, in memory of my uncle (Mohammad and Abdelhalim) and Aunt (Subhya, Fathya, and Ghazaleh) who passed away while I was doing my Ph.D. You will always remain in my thoughts, and I hope your souls will find peace in heaven.

To my parents-in-law Aisha and Abdeslam, for their encouragement and support.

Finally, to my dear sweetheart my wife Luna for the special and infinite support you provided me during the preparation of this manuscript.

With my best
Ahmad Aburomman

31 de enero de 2020

Resumen extendido

El crecimiento masivo de la automatización de procesos de negocio, así como la creciente adopción de las tecnologías de la información en su gestión, está produciendo una gran cantidad de datos de ejecución de procesos que se almacenan en forma de registros de eventos en los sistemas de información de las organizaciones. Aplicando técnicas de minería de procesos, es posible descubrir procesos reales ocultos, se pueden monitorizar los procesos existentes y también pueden mejorarse mediante el análisis de los registros de eventos.

Existen tres tipos principales de técnicas de minería de procesos: descubrimiento de procesos (*process discovery*), verificación de conformidad (*conformance checking*) y mejora de procesos (*enhancement*). Las técnicas de descubrimiento de procesos pueden abstraer un modelo de proceso sin utilizar ninguna otra información a priori, aparte de las trazas que componen los registros de eventos. Las técnicas de verificación de conformidad permiten comparar un modelo de proceso diseñado o definido previamente con el proceso real descubierto a partir de los registros de eventos, para mostrar dónde se desvía el proceso real del diseñado. La mejora del proceso tiene como objetivo ampliar o refinar un proceso existente, utilizando información relacionada con dicho proceso, que generalmente se extrae de los registros de eventos grabados.

En la mejora del proceso, se suele usar información temporal para medir los tiempos de espera entre las actividades del proceso, para verificar el comportamiento temporal durante la reproducción de trazas, para proporcionar información sobre problemas relevantes en el proceso (por ejemplo, cuellos de botella, tiempos de procesado, frecuencias) o para predecir el tiempo restante para la finalización de la ejecución de una instancia de un proceso. En este sentido, la predicción del tiempo restante de instancias de un proceso (casos en ejecución) ha sido referida en la literatura como uno de los desafíos actuales más importantes en la minería de procesos. El tiempo restante de una instancia de proceso es el tiempo requerido para que

dicha instancia finalice, desde su estado de ejecución actual. Predecir con precisión el tiempo restante es un tema clave para todos los actores que participan en la gestión de procesos de negocio. Para las organizaciones, tener predicciones de tiempo precisas les permite administrar adecuadamente los recursos, evaluar la calidad de los servicios que brindan o tomar decisiones administrativas apropiadas por adelantado. Para los usuarios finales, también es fundamental saber cuándo terminarán los procesos en los que están involucrados. Algunos ejemplos de esto último son los clientes bancarios que solicitan un préstamo, que pueden necesitar saber de antemano cuánto tiempo llevará revisar, verificar, evaluar y aceptar o rechazar su solicitud de préstamo, o los procesos de tratamiento médico, donde es crucial saber el tiempo restante de cada tratamiento para gestionar eficazmente los próximos tratamientos (por ejemplo, para preparar de antemano todos los recursos necesarios) o los siguientes pacientes.

El problema de predecir el tiempo restante es parte de un problema más general conocido como monitorización predictiva. En los últimos años, se han presentado varias propuestas centradas en monitorización predictiva y, más específicamente, en la predicción del tiempo restante. Inicialmente, estas propuestas se han centrado en la representación de las ejecuciones o trazas del proceso bajo la hipótesis de que las trazas con diferentes características tienen diferentes tiempos restantes. Varios de estos enfoques se basan en sistemas de transición anotados (ATS), donde cada traza en el registro de eventos está asociada a un estado que tiene una determinada representación. Otros enfoques utilizan una representación parcial basada en trazas o en índices. Más recientemente, se han propuesto enfoques para aplicar métodos de aprendizaje automático para la predicción del tiempo restante. En todos estos enfoques, sus respectivas codificaciones (es decir, las representaciones de trazas) incluyen información sobre el contexto del estado de ejecución del proceso, tal como la duración de las actividades o las variables de dominio, pero generalmente no incluyen información estructural relacionada con la ejecución de trazas que influye en la estimación del tiempo restante. Sin esta información sobre las características estructurales de las trazas, es difícil hacer predicciones precisas sobre el tiempo restante en casos complejos.

En esta tesis doctoral presentamos un nuevo enfoque basado en un ATS extendido con vectores que contienen características estructurales o atributos relacionados con la ejecución del proceso. En nuestro enfoque, cada estado del ATS está anotado con vectores que contienen información estructural sobre las trazas, como, por ejemplo, frecuencia de actividades, bucles de tamaño n, distancia de actividades y otros. Con base en estos vectores y los tiempos restantes de las trazas relacionadas con ellos, construimos un predictor basado en regresión lineal para cada estado, logrando así que la predicción tenga en cuenta la información estructural de las trazas. Hemos comparado los resultados de nuestro modelo con otras aproximaciones, utilizando diez conjuntos de datos de la vida real, y hemos obtenido predicciones más precisas que todas las propuestas del estado del arte.

Antecedentes

Recientemente, el enorme crecimiento de la automatización de procesos de negocio produce una enorme cantidad de datos relativos a la ejecución de dichos procesos. A partir de estos datos, las organizaciones pueden extraer y analizar información valiosa para descubrir, mejorar o cambiar sus procesos de negocio. Los sistemas de información que gestionan procesos extensos y cambiantes, almacenan toda la información relacionada con casos actuales y anteriores en forma de registros de eventos. Los registros de eventos no solo se usan para almacenar todos los datos generados por los procesos, sino que también se pueden usar más adelante para hacer que estos procesos sean visibles. Aplicando técnicas de minería de procesos, se pueden descubrir procesos ocultos ya que, según [83], el propósito de la minería de procesos es descubrir, monitorizar y mejorar los procesos reales mediante la extracción de información de registros de eventos fácilmente disponibles en los sistemas de información actuales.

En la bibliografía se han descrito diferentes tipos de modelos para predecir el tiempo restante de un proceso de negocio. Algunos de ellos usan representaciones basadas en estados donde las trazas del proceso (ejecuciones o instancias reales) se representan como una secuencia de estados y un conjunto de transiciones entre ellos: los estados modelan una secuencia de actividades de la traza, y las transiciones representan la ejecución de la siguiente actividad en la traza. Estas representaciones basadas en estados se denominan Sistemas de transición (TS). Cada estado de un TS se anota con información temporal sobre la ejecución del proceso, generando así un Sistema de transición anotado (ATS). La información contenida en la anotación se utiliza para predecir el tiempo restante mediante el uso de varias técnicas de estimación. En general, los resultados de predicción en estos modelos son modestos, principalmente debido a que se utiliza un número reducido de atributos utilizados para construir los modelos de estimación, pero también debido a las técnicas de estimación utilizadas. En general, estos modelos de estimación no incluyen información suficiente, dando lugar, en general, a una baja precisión en las predicciones incluso en casos no muy complejos.

Existen también otros enfoques en la bibliografía, que no siguen el modelo basado en ATS, y se basan en técnicas, como agrupamiento, redes neuronales y otros enfoques de aprendizaje automático [3]. Aunque estos modelos han mejorado los resultados de los modelos previos que siguen el modelo basado en ATS, tampoco los modelos enfoques no ATS no son capaces de lograr una alta precisión en las estimaciones [3].

Hipótesis

La hipótesis principal de esta Tesis es la siguiente:

- Las diferentes aproximaciones y modelos de la bibliografía de estimación de tiempo restante en procesos de negocio utilizan codificaciones de trazas que no representan explícitamente las complejas relaciones entre trazas; en particular, elementos estructurales como repeticiones de una misma actividad o de varias actividades, la co-ocurrencia de dos actividades cualquiera o la distancia entre dos actividades. Nuestra hipótesis de trabajo es que considerando la información estructural de las trazas, las estimaciones de tiempo restantes serán más precisas.

- Los modelos basados en Aprendizaje Automático han demostrado su validez para varias tareas relevantes en la predicción del tiempo restante, pero ningún enfoque ha considerado la información estructural sobre los rastros como discutimos en el punto anterior. Nuestra hipótesis a este respecto es que considerar la información estructural de las trazas en nuestro modelo superará a otros descritos en la bibliografía, incluidos los modelos de aprendizaje automático.

Objetivos

El propósito general de esta investigación es definir un nuevo modelo de ATS extendido basado en vectores que incluyan características estructurales relacionadas con la ejecución del proceso. En nuestro modelo, los estados del ATS se anotan con vectores que contienen información relacionada, por ejemplo, con la frecuencia de las actividades, repeticiones o bucles de actividades, distancia entre las mismas y otros. Nuestro modelo tiene como objetivo obtener estimaciones de tiempo restante para las trazas en los registros de eventos de procesos de negocio durante el tiempo de ejecución, a través de los siguientes pasos:

- Extracción y evaluación de diversas características en los registros de eventos, que proporcionen una caracterización estructural de sus trazas.

- Extensión del conocido modelo *Sistema de Transiciones Anotadas* (ATS) para incluir en él estas características.

- Aplicación de técnicas de regresión lineal para la predicción del tiempo restante de las trazas, para cada estado y combinación de características.

Metodología

La metodología que hemos seguido en esta tesis se basa en el método científico:

- Formulación de hipótesis iniciales, que en nuestro caso se realizó tras haber analizado los modelos existentes de predicción del tiempo restante en procesos de negocio y su precisión.

- Revisión del estado del arte, donde estudiamos el tipo de técnicas utilizadas para abordar los diversos objetivos, analizamos sus fortalezas y debilidades, y evaluamos otras técnicas diferentes que son útiles para abordar nuestros objetivos.

- La recopilación de observaciones en el alcance de esta tesis requirió la disponibilidad de datos para validar los algoritmos desarrollados.

- Diseño e implementación de los algoritmos que resuelven los objetivos abordados.

- Validación de nuestras propuestas en casos prácticos de uso de la vida real. Cuando la calidad de los resultados no es lo suficientemente buena, se lleva a cabo un proceso iterativo de modificación del algoritmo diseñado y su validación hasta que los resultados sean satisfactorios.

Contribución

Las principales contribuciones de esta Tesis se describen en los siguientes tres capítulos:

En el capítulo 4, describimos nuestra primera propuesta de un modelo que incluye información estructural de las trazas para predecir los tiempos restantes en un proceso de negocio (nuestro modelo básico inicial). Nuestro enfoque es una extensión del modelo ATS que denominamos EATS y consiste en:

- Definir ocho características relativas a los registros de procesos de negocio, que capturan información estructural de sus trazas.

- Ampliar el bien conocido modelo de sistema de transiciones anotadas (ATS) para anotar sus estados con los valores de las características anteriores para cada traza (atributos).

- Aplicar regresión lineal para predecir el tiempo restante del proceso para cada estado utilizando los valores de los atributos.

Las ocho características o atributos estructurales relacionadas con la ejecución del proceso de negocio están relacionadas con frecuencias, repeticiones, ciclos, etc. Cada uno de los estados en nuestro modelo se anota con una lista de vectores cuyos componentes son los valores de estos atributos para cada traza representada por dicho estado. Basándonos en estas listas y en los tiempos restantes de las trazas relacionadas con cada vector en la lista, construimos un predictor basado en regresión lineal para cada estado que considera la información estructural de las trazas.

Dado que nuestro modelo [2, 3] es una extensión del modelo base descrito en [88], nos centramos principalmente en el trabajo de comparación con dicho trabajo de referencia, que también es la base de muchos otros modelos descritos en la bibliografía. En los correspondientes capítulos de validación experimental, realizamos la evaluación empírica con diez conjuntos de datos conocidos de la vida real, mostrando que nuestro enfoque supera el modelo de referencia en las tres métricas consideradas (error absoluto medio, precisión y raíz del error cuadrático medio). Pero también hemos realizado experimentos para comparar este enfoque con otros modelos no ATS, entre ellos modelos basados en aprendizaje profundo. Los resultados muestran que nuestro modelo también supera a estos otros modelos. Sin embargo, en esta última comparativa, la desviación obtenida resulta alta, en general, lo que apunta a una dispersión de los resultados del modelo. Por lo tanto, estos primeros resultados de validación apuntan a que la inclusión de la información estructural produce mejores estimaciones del tiempo restante, pero que puede ser mejorada. Ello nos ha dado la base para continuar nuestra investigación tratando de refinar el modelo y así mejorar estos resultados iniciales obtenidos con el modelo básico.

En el capítulo 5, mejoramos nuestro modelo anterior al incluir una técnica de particionamiento en la lista de atributos asociada a cada estado del ATS, con el objetivo de obtener estimaciones de tiempo restante más precisas. En general, en el dominio de minería de procesos, los rastros en los registros de eventos de problemas reales suelen tener una gran variabilidad en términos de tamaño, número de actividades y tiempos de ejecución. Esta limitación es el escenario habitual para datos de procesos de negocio reales, como procedimientos o aplicaciones administrativas, gestión de incidentes industriales o procesos en un hospital u otras grandes organizaciones/instituciones. En muchos casos, el rango de valores de tiempo restante es muy amplio (y puede ir, por ejemplo, desde unos pocos segundos hasta varias horas) y esto incluso para trazas que son muy similares o incluso idénticas. Este es el caso también en los diez conjuntos de datos reales que hemos utilizado para la validación de nuestro modelo básico. Por lo tanto, las trazas en las listas pueden ser muy diversas y por tanto, en algunos casos reales, la precisión obtenida podría ser inferior a la necesaria. Adicionalmente, aunque esto ocurre tan solo en un número muy reducido de casos, el modelo básico podría ser superado por otros modelos en la bibliografía.

Para abordar este problema, en el capítulo 5, mejoramos el modelo básico descrito en el capítulo 4, añadiendo una técnica de partición de la lista de atributos asociada a cada estado del EATS. En primer lugar, motivamos este problema al presentar un ejemplo experimental simple que muestra cómo el modelo básico puede obtener valores bajos de precisión en algunos casos. Dicho ejemplo es un conjunto de datos de la vida real de una institución financiera relativo al proceso de solicitud para un préstamo personal. Los valores de precisión para este caso varían de 0,23 a 0,53, que son valores de precisión aceptables en el caso de la comparación con el trabajo de referencia, pero no lo suficientemente buenos para ámbitos de aplicación más exigentes.

En segundo lugar, definimos el mecanismo de partición y formalizamso cómo se calcula el tiempo restante en el nuevo modelo para cualquier nueva traza. Finalmente, validamos experimentalmente el nuevo modelo con partición, mostrando que realiza mejores estimaciones de tiempo restante comparado con los otros modelos en el estado del arte (no solo los modelos basados en ATS). La mejora de los resultados de estimación es un aspecto crucial, como hemos apuntado ya, porque a través de ella las organizaciones puedan realizar una gestión óptima de los recursos y también para mejorar la calidad de los servicios que brindan las organizaciones.

El enfoque que desarrollamos en el capítulo 5 consiste en:

- extraer y evaluar nuevamente las ocho características en los registros de procesos de negocios, que proporcionan una caracterización estructural de los rastros
- extender el conocido modelo de sistema de transición anotado (ATS) para incluir estas características, dando lugar al nuevo modelo EATS

- aplicar una técnica de partición de las listas asociadas a cada estado del EATS, para agrupar las trazas con características similares
- aplicar una técnica de regresión lineal a los atributos descritos en cada una de las particiones, para mejorar la predicción del tiempo restante de las nuevas trazas

Nuestro método de partición tiene como objetivo dividir la lista de vectores asociada a cada estado del ATS en varias particiones, cada una de las cuales contiene todas las trazas parciales que tienen tiempos restantes similares. La similaridad en este aspecto se define mediante un valor de umbral de la siguiente manera: una vez ordenada la lista por tiempos restantes, se considera que dos trazas consecutivas pertenecen a la misma partición si el cociente de sus tiempos restantes está por encima de un umbral predefinido (es decir, sus tiempos restantes se consideran similares) De lo contrario, si el cociente está por debajo del umbral, se agruparán en diferentes particiones, puesto que sus tiempos restantes se consideran diferentes. Una cuestión importante en este nuevo modelo mejorado de partición es la selección del umbral, ya que define tanto el número como el contenido (nivel de similaridad entre las trazas) de las particiones en las listas. Las particiones se asocian a intervalos de tiempo restante que pueden ser o bien intervalos muy cercanos o muy amplios en función del umbral escogido. Esto debe tenerse en cuenta para definir un número de particiones equilibrado, para que no sea muy bajo, ya que los valores de rango para el tiempo de estimación serán altos y, en principio, la precisión menor, pero tampoco muy alto, ya que los rangos serán menores, la precisión mayor (con riesgo de sobreajuste) y un mayor coste computacional del proceso. A este respecto, describimos en la tesis un método para calcular un valor de compromiso para el umbral, que no depende de un caso particular. Para este cálculo, utilizamos el conocido método ”regla de un error estándar” para la selección de modelos [38], que muestra empíricamente que este compromiso puede lograrse, dando lugar a buenas estimaciones de tiempo restante que superan el rendimiento todos los otros modelos descritos en la bibliografía.

Después de dividir las listas, aplicamos regresión lineal para cada partición para así obtener un modelo de predicción. Para predecir el tiempo restante de una nueva traza, buscamos la partición a la que pertenece esta nueva traza y luego le aplicamos la expresión de regresión lineal correspondiente. Con respecto al resultado, la validación realizada con diez conjutos de datos reales, mostró que el modelo con particionamiento mejoró la precisión no solo en comparación con el trabajo de referencia [88], algo ya logrado con el modelo básico, sino también con todos los restantes trabajos descritos en la bibliografía [97], para todas las métricas consideradas. Adicionalmente, la desviación estándar de los resultados, se mejora muy considerablemente frente al modelo básico.

En el capítulo 6, presentamos una mejora en el mejorado el modelo con particionamiento, para abordar su escalabilidad, mediante la agregación de una etapa de selección de atributos previa a la aplicación de la regresión.

La cuestión de la escalabilidad surge del hecho de que, aunque los resultados del modelo anterior mejoran los del resto del estado del arte, el número total de atributos a considerar puede llegar a ser elevado en aquellos procesos de negocio cuya cantidad de eventos sea muy alta. Ello puede derivar en problemas de coste computacional del modelo en dichos casos. Los motivos para abordar esta cuestión son principalmente dos: en primer lugar, el número de particiones creadas en cada estado, que aumenta en gran medida a medida que aumentan el número de actividades y se pudieran considerar valores umbral muy altos. En segundo lugar, la cantidad de atributos, que en algunos casos es lineal y en otro cuadrático con el número de eventos de la traza.

Por lo tanto, para resolver los problemas mencionados anteriormente, en el capítulo 6 hemos introducido dos métodos clásicos para realizar la selección de atributos: *i*) *Forward Best-First*, que sigue una estrategia voraz *hill-climing* de incremento del número de atributos (partiendo de cero), combinado con una estrategia vuelta atrás y *ii*) *Forward Greedy Stepwise*, que realiza una búsqueda voraz (hacia adelante o hacia atrás) utilizando el espacio de subconjuntos de atributos.

El objetivo es disminuir el número de atributos (y, por lo tanto, reducir el número de operaciones y el costo computacional del método) manteniendo la precisión de predicción del método dentro de límites aceptables. Tal y como se muestra en la experimentación realizada, sobre los mismos diez conjuntos de datos reales, nuestro modelo proporciona un modelo equilibrado que produce un tiempo de predicción restante aceptable y un menor consumo de tiempo. Así, nuevamente en todos los casos este modelo mejora los resultados del modelo base descrito en [88].

Contents

CHAPTER 1

INTRODUCTION

In recent decades, there has been great interest in introducing and developing new techniques, models and systems to automate the processes of organisations and enterprises. This has caused the data and information registered by the information systems in organisations to increase considerably, which has led to organisations and enterprises paying further attention to these data and considering the best techniques for extracting the relevant information.

In this regard, the massive growth of business processes automation, as well as more technology adoption in business process management, is producing a vast amount of process execution data that can be stored in the form of event logs [71, 95]. By applying process mining techniques, hidden processes can be discovered [63, 66], since 'Process mining aims to discover, monitor and improve real processes by extracting knowledge from event logs readily available in today's information systems' [88]. There are three main types of process mining techniques [82]: process discovery, conformance checking, and process enhancement. A discovery technique takes an event log and produces a model without using any a priori information. Conformance makes a comparison between an existing process model and the process obtained from the event log, aiming to show where the real process deviates from the modelled one [84]. Finally, in process enhancement, temporal information is used in different ways, such as to measure the wait times among process activities, to check the temporal behaviour during traces replay, to provide information about bottlenecks, through times, frequencies, etc. or to predict the remaining times for running process instances [84].

This thesis is framed around the latter concept, since here we are proposing a new model for obtaining more precise time estimations of business processes. Therefore, our work belongs to the area of Business Process Enhancement within Business Process Mining techniques.

1.1 Motivation

Predicting the remaining time for process instances is highlighted as one of the most important challenges in process mining [82]. The remaining time of a process instance is the required time for a process instance to be finished from a particular execution state of this instance. Predicting this time is a key point for organisation, since it allows them to move from one state to another taking into account optimal management of the resources [8]. It is not only crucial for the quality of service of organisations, but also for the end-user to be comfortable with how long it will take to finish [63]. In addition, from the customers' point of view, the completion time of processes is critical [71]. For example, bank customers applying for a loan need to know how long it will take for their loan application to be reviewed, checked, assessed and accepted or declined. In other fields, such as healthcare processes, it is crucial to know the remaining time of each treatment in order to effectively manage the next treatment and prepare all the necessary resources (e.g. special equipment, ICU, operating room, etc.).

1.2 Hypothesis

After studying state-of-the-art methods in the prediction of process remaining time, our hypothesis is that **the structural information in the traces of a business process provides relevant information for achieving accurate remaining time predictions** and that **there are no models in the literature which consider structural information in their estimations**. We have built on our work as we believe that this valuable information extracted from the events log will enhance the prediction of the remaining time of a process.

1.3 Objectives

The desired goals of this dissertation involves what we hypothesised at the beginning of this research, which encourages us to exploit business processes and process mining to define a model-based methodology for Remaining Time Prediction in Business Processes using machine learning techniques in terms of:

1. The feasibility of using process mining techniques to extract valuable information from the tasks of the process. Process enhancement aims to extend or improve an existing process, using information related to the process recorded in some event logs. By extracting timestamps from the event log and extending it in the process model, it is possible, for instance, to measure the wait times between process activities. Timestamps can also be used to check the temporal behaviour during replay. Time differences between related activities can be used to add predictable wait times to the model. In our work, using timestamps in the event log can enhance the model with the complete time for each activity.

2. The feasibility of using machine learning techniques to predict the remaining time for each activity in a process.

1.4 Contribution

The objective of this thesis is to define a new process mining enhancement model which will extract different attributes from the process logs that include the structural information of such processes. Our approach adds to the well-known baseline Annotated Transition Model [88], endowing it with a new remaining time estimation model which significantly improves the already state-of-the-art methods by taking into account the structural information of the traces.

1.5 Thesis Outline

In Chapter 2, we introduce the background of the thesis and some fundamental concepts such as Business Process Management or Process Mining, among others. In Chapter 3, we review the Related Literature in this field and present the most relevant theory related to remaining time prediction in business processes. In Chapter 4, we define and validate our remaining-time estimation base model which adds to the ATS-based model described in [88]. In Chapter 5, we add a partitioning technique to the base model which allows us to improve the estimations, outperforming all the models in the literature. In Chapter 6, we present, discuss and analyse several approaches to address the scalability of our model. Finally, in Chapter 7, we summarise the thesis, present the most relevant conclusions and state an overview of unresolved issues, namely the limitations that constitute the next research steps and future work.

CHAPTER 2

BACKGROUND

In this chapter, we are going to introduce the basic topics and concepts needed for explaining our model. We aim to explain some terms about Business Process Management in Section 2.1; Process Mining in Section 2.2, and Transition Systems, in Section 2.3. We will provide an overview of different process modelling techniques, but we will describe the Transition System in detail as this is the process modelling technique that we have worked on and used in our work.

2.1 Business Process Management

The business process management concept understands every product or service in an organisation as the result of a sequence of activities. Business processes are the main tool used for organising these activities and achieving a better understanding of the relationships between them [100]. Information Systems and, in general, Information Technology, have an important role in business process management, since information systems drive the number of activities generated in any organisation. Business process activities can be accomplished by employees manually or through the support of information systems. Conversely, some business process activities can be done automatically by systems or services, without any human intervention [99].

Any company can effectively achieve its business objectives by having employees and information systems resources cooperate in a compatible way [36]. Therefore, the role of business processes is important to facilitate effective cooperation whilst trying to avoid inconsistencies among the organisational aspects of business and information technology [102].

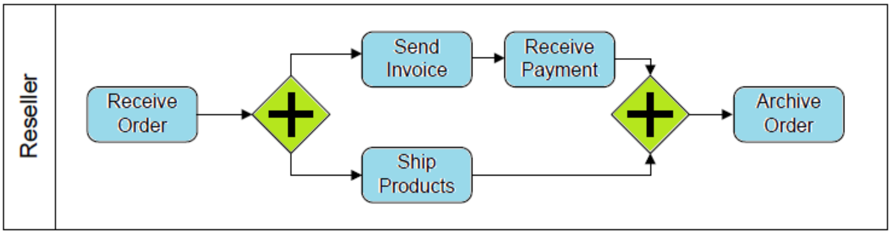


Figure 2.1: Simple ordering process of re-seller (adapted from (100)).

Working towards bridging the gap between management and technology is extremely important because current markets are very dynamic, meaning that these companies must remain competitive when offering their products and services to satisfy customers. Narrowing the gap between regulation and technology is essential because these dynamic markets are forcing companies to consistently provide better and more specific products to their customers [45]. For instance, successful and bestseller products today will not be the same tomorrow. If other competitors offer the same product at a cheaper rate, with a high-quality design, or one that is more comfortable to use, the first product will probably see a decrease in sales compared to the second product on the market.

The methods and technologies of various scopes of business administration and computer science have affected business process management. Considering the early start and its three decades of business organisation and management, the role of process orientation has risen and with it a new way of organising and managing companies was proposed [86, 24].

Based on the different descriptions of business processes, we will adopt the following definitions:

Definition 1 (Business Process) [100]: A business process consists of a set of activities that are performed collectively in an organisational and technical environment. These activities work together to achieve a business goal. A single organisation enacts each business process, but it may also interact with business processes belonging to other organisations.

Once we understand and define the business processes, their components and their interrelations, the scope is then expanded to include the concept of Business Process Management, which represents not only business processes but also additional activities.

Definition 2 (Business Process Management, BPM) [100]: This notion includes concepts, methods and techniques to support the design, administration, configuration, enactment, and analysis of business processes.

In Figure 2.1 we illustrate a simple example of the ordering process of a re-seller company. Activities in the process are represented by nodes and their flow by arrows. The process includes sequential activities (‘send invoice’, ‘receive payment’), but also parallels (‘ship products’ alongside ‘send invoice’ and ‘receive payment’). In this ordering process, first, an order is received, then an invoice is sent, and afterwards the payment is received; in parallel to this, the ordered products will be shipped. Finally, the order is archived. Notations are used to express the organisation of activities of a business process.

Most companies and organisations try to become more efficient and effective in the context of worldwide competition [74, 10]. In order to achieve this, some tools and mechanisms have been created to aid in the management of business processes. Business Process Management can be characterised through the business process lifecycle [54].

2.1.1 Business Process Lifecycle

The business process lifecycle [89] provides a high-level view of business processes, in addition to producing a description of the phases of a business process. The lifecycle has several phases: (1) Evaluation, (2) Design and Analysis, (3) Configuration and (4) Enactment of business processes [100], as shown in Figure 2.2.

In the first step, the process evaluation phase, the business processes that take place within a company are assessed. To achieve this, some tools and approaches can be applied, for example process mining [41], to understand how the company’s processes are running. In the next step, the design and analysis phase, the processes are recognised. In addition, specialists recognise these processes in the form of business process models. After these models are created, they can be edited and redesigned [52]. These redesigned models are then assessed for accuracy, using tools that simulate these models to test if the modifications introduced are genuinely improving the processes [15].

In the configuration phase, as soon as the business process model is chosen for adoption, the next step is to implement the process in the process environment. This will depend on where and how this phase takes place. For example, employees may be told to follow the new specifications or to modify the workflow engine [86] in order to automatically control the business process execution as planned, although this only happens if the type of business is allowed to be automated. Some industries, such as health care, face a big challenge when trying to implement this [86].

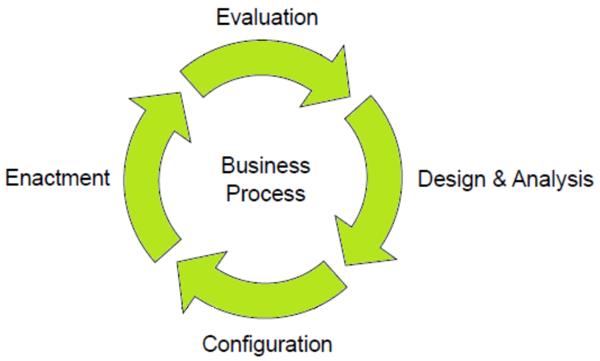


Figure 2.2: Business process lifecycle. (Adapted from (100)).

Certain cases, where these processes can be understood in a good way and show little variance and a high number of occurrences, would be best suited for automation by workflow engines. Nevertheless, for some process enactment environments, we consider using workflow engines to be pointless. This infeasibility can be explained by the huge demand for flexibility or complicated or unusual cases. The Enactment phase, which is the last step, includes the real run-time of the business processes, which are stimulated to accomplish the organisation's objectives. Process initiation usually happens depending on the initiating event, such as receiving a query sent by a client.

2.1.2 Execution and Monitoring

The process enactment phase is where the organisation or enterprise achieves their daily business. In this step, everyday aspects of running a company are taken into account. Clients buy goods or services from a business, who will be in charge of delivering that product by working through the business processes. The business process management system efficiently controls the execution of business process cases, in the same way as declared in their model. The system also ensures the same performance of the process activities within the same execution boundary in the process model. To do so, process enactment needs to have the correct process composition.

The most important part of the execution of business processes, from the management point of view, is whether the processes are carried out efficiently, i.e. carried out over a short period and with fewer economic resources, and whether its execution fits the created model. Some techniques can be provided by business process analytic [105], which help with operational tasks, for example, the detection of bottlenecks.

The monitoring component is used by the business process management system to visualise the status of business process instances. Process monitoring uses a crucial mechanism to provide precise and reliable information about the status of business process instances. The provided information can be used, for instance, to improve the processes or to convey an answer to a client inquiry. To conduct business process analytic, we need to be able to measure the performance of a business process. This task is called process monitoring. Its objective is to allow for informed decisions during the enactment phase as mentioned in [15]. High-quality performance of the process can only be achieved if we can provide information about how the process ran. As a result, it is necessary to measure the process. The monitoring phase allows us to react appropriately to modifications that might occur during the process, such as an increase in the demand of a product. This means that, if we see that many cases have arrived, but not many of them are completed, then we can understand that the number of cases is growing. To deal with this situation, business managers might, for instance, put resources into use (hiring new specialists) to solve the problem with increasing demand and so that clients do not have to wait for a long time for their product or service to arrive [86].

2.1.3 Business Process Models

Business process models are the central hub for implementing different business processes. These are the most important ones in companies and enterprises that capture their business and then generate products or services to their customers [100]. Creating business processes models is crucial for managers in business process management because these models allow us to collect the essential components of a business process before describing the relationships between them and the order of execution for each part.

According to Indulska et al. [42], there are five crucial advantages of business process modelling: 1) process improvement, 2) understanding, 3) communication, 4) model-driven process execution and 5) process performance measurement. It is clear that every company will try to get to these advantages, and of course, these models should be designed to achieve excellent quality [55] and produced by specialists who know how to do it using appropriate modelling language.

There are indeed some tools for modelling, which focus on capturing business processes [47]. For instance, flowcharts have been used to model algorithms since the beginning of the computing era, together with other tools like UML activity diagrams [62], event-driven process chains or Business Process Modelling Language and Notation (BPMN) [61], Petri Nets (PN) or Transition System (TS) [80]. There are many other process models, all of which are useful when organising business process models, but in this thesis, we will use the transition system (TS) as our base process model. More details about the TS and our contribution based on it will be described in Section 2.3 .

2.2 Process Mining

Process Mining is a popular research topic in the interaction between machine learning and data mining, on the one hand, and process modelling and analysis on the other[82]. The purpose of process mining is to discover, monitor and improve real processes by extracting data and information from real-life event logs that exist in daily systems [80, 82, 67].

It is well-known that information systems play a major role in the corporation's and enterprise's business. As a result, millions of events are collected every day by information systems. However, corporations and enterprises have challenges and sometimes face a problem when extracting data from these events. Process mining appears to have overcome these problems, its goal being to extract business processes from the event data, for instance, by discovering the process model of any event log recorded by corporations and enterprises. The importance of process mining increases with the growth of the event log and the need to translate the event data into process models [88].

In this project, we will assume that business process models can be easily found, and we also have an accurate representation of how a business process works. If these models are non-existent, this should not be considered to be a limitation, since process mining techniques do exist and can be applied to uncover business process models from past events as described in these works [63, 66, 69].

2.2.1 Process Mining Categories

The process mining manifesto [82] considers that process mining is centred around four components: the real world, process models, software systems and event logs, as described in Figure 2.3, The existence of the event log allows for the derivation of the process model, and the conformance technique regularly checks the reality of the model and ensures the compliance of both model and log. The enhancement technique aims to change/extend the existing

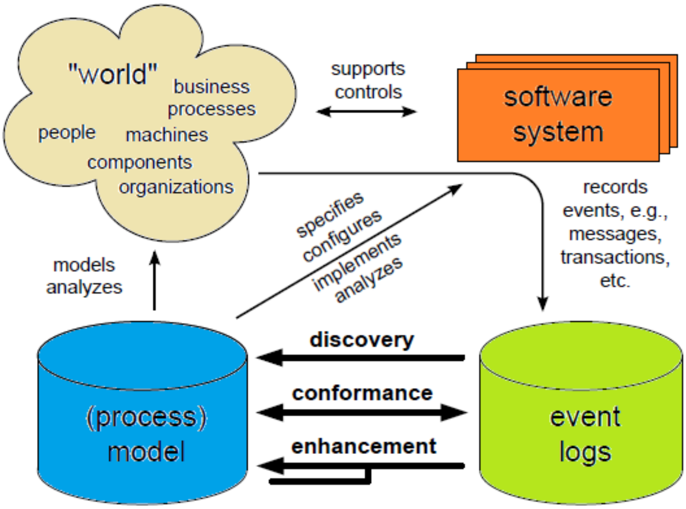


Figure 2.3: Process mining overview. The three major aspects of process mining are 1) discovery, 2) conformance, and 3) enhancement. (Adapted from (86)).

model by enriching it with information from its log. Process mining techniques are classified into three categories:

1. Discovery, in which a model can be derived from event logs that provide information about traces of processes and how they are executed.
2. Conformance, in which a process model is compared to the actually executed event logs.
3. Enhancement, in which a process model is improved and enriched with extracted data from the event logs.

We will briefly describe these aspects before describing them in more detail, as Enhancement is the core method we used in our method.

Discovery of Process Models

Problematising the issue of discovery (in other words, recognising the minimum process modelling which provides insight into how elements are related and seen by the event logs) has a direct relationship with the issue of discovering how minimum finite-state automatons are somewhat compatible with the information, as described in [34]. In fact, this is a computational challenge; apparently, this issue has motivated general interest in research and development fields because it promises to easily automate the task of manually extracting process models.

Agrawal et al. were the first to explore and research the process mining field [5], They described how to automatically extract models from events that were stored in an event log. Since then, mining algorithms appeared, such as Algorithmic approaches, for instance, the alpha algorithm [91], the one that discovers ‘Petri nets’, or the related work by Herbst and Karagiannis [39]. A model was built by the latter which reflects all traces as a Markov chain and simplifies the model by iterative combining phases to make the design simpler. Moreover, to solve this problem, some techniques using heuristic approaches, for instance, genetic algorithms, were used [21].

Conformance Checking

To make sure that a business process achieves the best quality possible, it is not enough to only specify how the processes should be executed. It is also crucial to evaluate whether the given models show the same behaviour of the sequencing process and are carried out as was registered.

When talking about conformance checking, we refer to the field within process mining, which checks if the actual execution of a business process, as recorded in the event log, conforms to the model and vice versa. It assists in recognising the parts designed in the process model that need to be enhanced, or in which workers would have to modify their work, in order to fit the model. Rozinat and van der Aalst [72, 73] were the first to apply a way of replaying an event log in a model and calculating the number of further inserted and removed tokens that are required to replay a trace in a Petri-net model.

In their work, they found that the traces in an event log are excessively replayed in a model, therefore there is no guarantee of discovering the optimum route through the model. Also, in another subsequent work, the notion of cost-based alignment was introduced by Adriansyah et al. [4]. In subsequent projects they work to find global optimal alignment [4], which takes into consideration the properties' structure of the model. Therefore, it defines the costs to additional steps in either model or trace that cannot be mimicked by the other.

Fitness measurements are used to evaluate the conformance measurement [72]. As a result, the fitness value can tell us about how many degrees the observed cases in the event log keep track of the behaviour specified by the model, taking into account that the fitness value calculated by fitness measure ranges between 0 and 1.

Enhancement of Process Models

The third category of process mining is concerned with the problems that arise when we have a combination of any process model and an event log as input. At this stage, the mission is to enrich these models using the information that exists in the event logs. Enhancement techniques provide us with the ability to make these challenges (bottlenecks, service levels, throughput times and frequencies) visible [82, 83].

In some cases, some process models do not fit the event logs completely, which could happen due to extraordinary cases which are not contained while generating the model, but were executed and recorded in reality. Therefore, this stage can provide users with the ability to adjust business process models automatically in the best way to reflect the better-observed event logs [9, 26].

It is also possible to extend or improve an existing process model using the event log. A non-fitting process model can be modified and corrected using the available diagnostics provided by the alignment of the model and event log. It is feasible to enrich an already designed process model using an event log. In that way, if there is a non-fitting process model, then at this stage these non-fitting models can be corrected by the diagnostics of the alignment technique.

In Figure 2.4, the three types of process mining defined in terms of input and output, namely discovery (a), conformance checking (b) and enhancement (c), are shown. According to Figure 2.4, discovery techniques take an event log as input and produce an output model. In general, the discovered model is a process model, such as a Petri net, a BPMN, an Event-driven Process Chain (EPC) or a UML activity diagram. Nevertheless, this discovered model may also take into account some other perspectives, such as, for instance, a social network. Conformance-checking techniques take not only an event log but also a model as input, while their output consists of a diagnostic report about the similarities and differences between the log and the model. Finally, techniques for model enhancement, either by repairing or

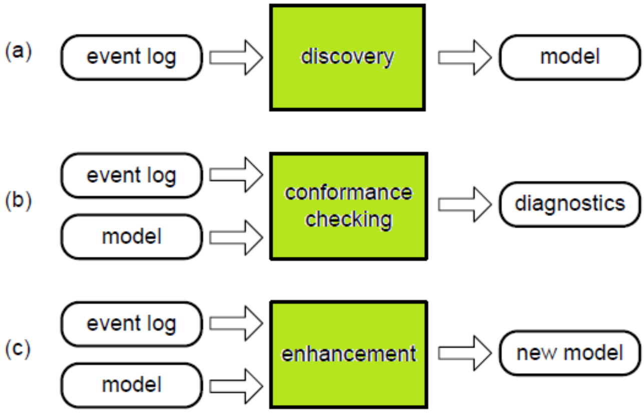


Figure 2.4: The three basic types of process mining explained in terms of input and output: (a) discovery, (b) conformance checking, and (c) enhancement. (Adapted from (82)).

extending it, also take an event log and a model as input, their output being a new improved and extended model respectively [23, 82].

In addition, process mining should cover different perspectives, namely control-flow, organisational, case and time perspectives. The control-flow perspective is focused on establishing the order of the activities, in the sense of finding a good characterisation of all possible paths. The result of mining this perspective is typically expressed in terms of a Petri net or some other process notation, such as EPCs, BPMN, or UML activity diagrams. The organisational perspective is focused on analysing information regarding the resources hidden in the log, studying which actors (people, systems, roles or departments) are involved and how they are related [82].

In this way, this mining perspective claims either to structure the organisation by classifying people in terms of their roles and organisational units or to show the social network. The case perspective is focused on the characteristics of the cases. In general, the main characteristics of the cases are their path in the process and the actors working on them. Nevertheless, cases can also be characterised by the values of the corresponding data elements. For instance, for a replenishment order case, it may be interesting to know the supplier or the number of ordered products. Finally, the time perspective is focused on the analysis of the timing and frequency of the events, making it possible, whenever events bear timestamps, to discover bottlenecks, measure service levels, monitor resources usage and predict the remaining processing time of running cases [23].

Finally, it is worth mentioning that there have long been some misconceptions regarding process mining. For instance, it is the typical case that some vendors, analysts and researchers limit its scope to a special data mining technique for process discovery that can only be performed offline. Fortunately, this is not true, and it is important to highlight the following three characteristics of process mining:

- **Process mining is not limited to control-flow discovery**. Although both practitioners and academics usually consider control-flow discovery as the most exciting part of process mining, process mining is far from being limited to this. On the one hand, as can be seen from Figure 2.4, discovery is just one of the three basic forms of process mining, together with conformance and enhancement. On the other hand, the scope of process mining is not limited to the control-flow perspective, it being also important to consider the organisational, case and time perspectives [82].

- **Process mining is not just a specific type of data mining**. On the contrary, it can rather be considered as the appropriate link between data mining and traditional model-driven Business Process Management (BPM) [80]. In fact, since most of the available data mining techniques are not process-centric, process models, which tend to exhibit concurrency, are not comparable to simple data mining structures such as decision trees and association rules. In this context, it is necessary to resort to entirely new types of representations and algorithms to achieve that aim.

- **Process mining is not limited to offline analysis**. although process mining techniques are based on the extraction of knowledge from historical event data [69]. Even when using ‘post-mortem’ data, the obtained results can be applied to running cases. For instance, the completion time of a partially handled customer order can be predicted by resorting to a discovered process model [82].

2.2.2 Events and Event Logs

Events and event logs are an important concept in the core part of process mining. Event logs are the main part of process mining because they include the target information as raw data. Events can be generating by different sources, like information systems or enterprise resource planning systems.

Even though the term event is not exactly clear as in can be used to various situations with a different significance, that is, in logistics, an event means something very important that requires for a modification in a plan [33]. Luckham defines an event as something that has significance and influences a specified system [48].

Definition 3 (Event) [88]: An event, e, is described by a unique identifier and is characterised by its properties, such as its identifier, timestamp and the activity which is executed in the timestamp.

In this chapter, we will understand an 'event' to be a positive landmark concerning business processes, in particular, events that show that the process to which it is related has achieved its goal and changed its state, for instance, from 'start' to 'complete'. We will not delve into the specific terminology of how events are sourced. As a result, we will only take into account event logs that provide important information about the business process, as can be found in process mining [41].

Definition 4 (Trace, Event Log) [88]: A trace, T, is a sequence of ordered events $\{e_1, e_2, \ldots, e_M\}$. An event log is a set of traces.

Considering that event logs do not just relate to the components in the definition we have provided above, many information systems still keep track of events' characteristics such as data, costs and resources. Nevertheless, in this thesis we will consider the time aspect between the process instances. In a real setting, we consider the direct relationship (correlation) between the mentioned events and cases to be important. We are not going to explain how problems with correlation are dealt with. If readers would like to know more about the topic, they can refer to Agrawal et al. in [18]. Motahari-Nezad et al. [59] also introduce a new perspective, as they provide correlation conditions that control how events will correlate with activities and offer the discovery capability of these conditions from an event log. On the other hand, Musaraj et al. [60] provide a methodology that identifies the correlation on a message level based on the timestamps value.

Our assumption is that the environment is the factor responsible for events and the factor that also generates the timestamps (i.e. each event with its own timestamp) and domains are the factors that set the particular timestamps. However, in some cases, timestamps can also create the event (e.g. at a specific time, event 'X' should start). In other cases, timestamps could even be extracted from the event sources.

2.2.3 Analysing an Example Log

After providing an overview of process mining and positioning it in the broader BPM discipline, we will use the event log shown in Table 2.1 to clarify some foundational concepts. The table shows just a fragment of a possible log corresponding to the handling of requests for compensation. Each line presents one event. Note that events are already grouped per case. Case 1 has five associated events. The first event of Case 1 is the execution of the activity register request by Pete on 30 December 2010. Table 2.1 also shows the unique ID for this event: 35654423. This is merely used for the identification of the event, for example, to distinguish it from event 35654483 that also corresponds to the execution of the activity register request (which is the first event of the second case). Table 2.1 shows a date and a timestamp for each event. In some event logs, this information is more general and only a date or partial ordering of events is given.

Table 2.1: A sample of event log

Case id	**Event id**	**Properties**				
		Timestamp	Activity	Resource	Cost	. . .
1	35654423	30-12-2010:11.02	Register request	Pete	50	. . .
	35654424	31-12-2010:10.06	Examine thoroughly	Sue	400	. . .
	35654425	05-01-2011:15.12	Check ticket	Mike	100	. . .
	35654426	06-01-2011:11.18	Decide	Sara	200	. . .
	35654427	07-01-2011:14.24	Reject request	Pete	200	. . .
2	35654483	30-12-2010:11.32	Register request	Mike	50	. . .
	35654485	30-12-2010:12.12	Check ticket	Mike	100	. . .
	35654487	30-12-2010:14.16	Examine casually	Pete	400	. . .
	35654488	05-01-2011:11.22	Decide	Sara	200	. . .
	35654489	08-01-2011:12.05	Pay compensation	Ellen	200	. . .
3	35654521	30-12-2010:14.32	Register request	Pete	50	. . .
	35654522	30-12-2010:15.06	Examine casually	Mike	400	. . .
	35654524	30-12-2010:16.34	Check ticket	Ellen	100	. . .
	35654525	06-01-2011:09.18	Decide	Sara	200	. . .
	35654526	06-01-2011:12.18	Reinitiate request	Sara	200	. . .
	35654527	06-01-2011:13.06	Examine thoroughly	Sean	400	. . .
	35654527	06-01-2011:11.43	Check ticket	Pete	100	. . .
	35654527	06-01-2011:09.55	Decide	Sara	200	. . .
	35654527	06-01-2011:10.45	Pay compensation	Ellen	200	. . .

In other logs, there may be more elaborate timing information, including when the activity was started, when it was completed, and sometimes even when it was offered to the resource. The times shown in Table 2.1 should be interpreted as completion times. In this particular event log, activities are considered to be atomic and the table does not reveal the duration of activities. In the table, each event is associated with a resource. In some event logs, this information will be missing.

In addition, in other logs, more detailed information about resources may be stored, for example, the role that a resource has or elaborated authorisation data. The table also shows the costs associated with the events. This is an example of a data attribute. There may be many other data attributes. For instance, in this particular example, it would be interesting to record the outcome of the different types of examinations and checks. Another data element that could be useful for analysis is the amount of compensation requested. This could be treated as an attribute of the whole case or stored as an attribute of the register request event.

Table 2.1 illustrates the typical information present in an event log. Depending on the process mining technique used and the questions at hand, only part of this information will be used. The minimal requirements for process mining are that any event can be related to both a case and an activity and that events within a case are ordered. Hence, the 'case id' and 'activity' columns in Table 2.1 represent the bare minimum for process mining. By projecting the information in these two columns, we obtain the more compact representation shown in Table 2.2. In this table, each case is represented by a sequence of activities also referred to as a trace. For clarity, the activity names have been transformed into single-letter labels, i.e. a designated activity register request.

Table 2.2: Process model usual representation, where the activities above described in Table 2.1 are labelled as follows: a = register request, b = examine thoroughly, c = examine casually, d = check ticket, e = decide, f = reinitiate request, g = pay compensation, and h = reject request

Case id	Trace
1	<a, b, d, e,h>
2	<a, d, c, e, g>
3	<a, c, d, e, f, b, d, e, g>

Process mining algorithms for process discovery are the main tool used for converting information like the one shown in Table 2.2 into process models. For example, the basic α-algorithm [91] represents the discovered model with a Petri net. Figure 2.5 shows an example

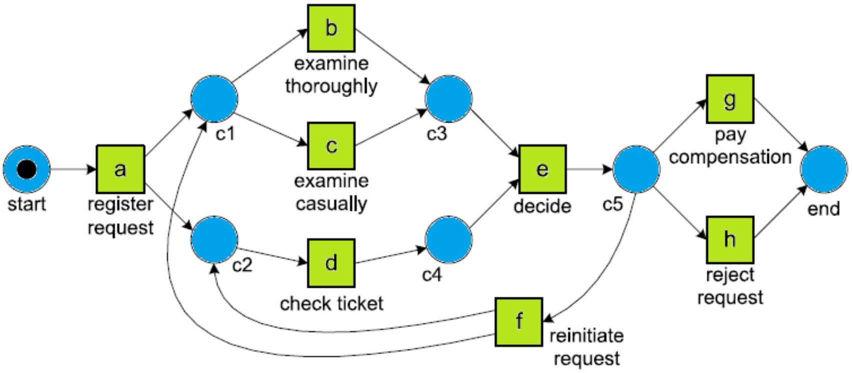


Figure 2.5: The process model discovered by the α-algorithm (91) based on the set of traces shown in Table 2.2. (Adapted from (80)).

of the model obtained for the data in Table 2.2. We can see that all the three traces in Table 2.1 exist in the process model. The trace of the first case, <a, b, d, e, h>, is used as an example to show that the trace 'fits' or in other words 'conforms to' that model.

In Figure 2.5, the initial marking 'a' is indeed enabled, because of the token in 'start'. After firing 'a', places 'c1' and 'c2' are marked, i.e., both places contain a token. 'b' is enabled at this marking and its execution results in the marking with tokens in 'c2' and 'c3'. Now we have executed <a, b>, and the sequence <d, e, h>, remains. The next event 'd' is indeed enabled and its execution results in the marking enabling 'e' (tokens in places 'c3' and 'c4'). Firing 'e' results in the marking with one token in 'c5'. This marking enables the final event 'h' in the trace. After executing 'h', the case ends in the desired final marking with just a token in place end. In a similar way, the other two traces in Table 2.2 (<a, d, c, e, g>, <a, c, d, e, f, b, d, e, g>) can be identified.

2.2.4 Play-in, Play-out, and Replay

One of the most important aims of process mining is to accurately establish the relationship between a process model and the reality reflected by the event logs. According to the terminology first introduced by David Harel in the context of Live Sequence Charts [37], this relationship can be described in terms of Play-in, Play-out and Replay. These three concepts, shown in Figure 2.6, are described below.

Play-out is generally associated with the classic application of process models, in the sense that behaviour is generated based on an input model. For instance, within the context of Play-out, the traces shown in Table 2.2 can be obtained by using the Petri net depicted in Figure 2.5 for repeatedly playing the token game. In particular, Play-out is focused not only on the analysis but also on the enactment of business processes. In fact, a 'Play-out engine' can work as a workflow engine capable of controlling cases by letting them only make the movements that are allowed by the model, enacting, in this way, the operational process based on the executable model. In addition, Play-out engines can also be used to carry out experiments within the context of several simulation tools, allowing them to collect valuable statistical information, such as confidence intervals, by running the model repeatedly. Here, it is important to note that the feasibility of this Play-out application relies on the fact that, although simulation engines interact with modelled environments while workflow ones interact with the real actors, such as workers and customers, both engines are quite similar. Finally, the Play-out approach can also be associated with the so-called model checking approach, which is a traditional verification method based on exhaustive state-space analysis [17].

Play-in, often referred to as inference, is the opposite of Play-out, in the sense that a model is built based on a given behaviour. In this case, contrary to the Play-out case, the Petri net depicted in Figure 2.5 can be automatically inferred from a given event log, for instance, the event log in Table 2.2. Different Play-in techniques, such as the α-algorithm and other process discovery approaches, have been proposed in the process model field. In addition, the Play-in approach is widely used in the field of data mining since techniques within this field also build models based on available samples. Nevertheless, since data mining has not been long concerned with process models, most of the traditionally used data mining techniques are not suitable for Play-in process models. Fortunately, in order to fill this gap, in recent years researchers have developed some process mining techniques capable of discovering process models based on event logs.

Replay, for its part, takes both an event log as well as a process model as input. The replay approach is then based on 'replaying' the event log on top of the process model. For instance, in the Petri net depicted in Figure 2.5, the trace <a, b, d, e, h> can be replayed by 'playing the token game', i.e. by setting the transitions to fire in the indicated order. There are different reasons for replaying an event log:

- Conformance checking: Replaying the log can help to highlight, detect and quantify the existing conflicts between the log and the model. For instance, if the trace <a, b, e,

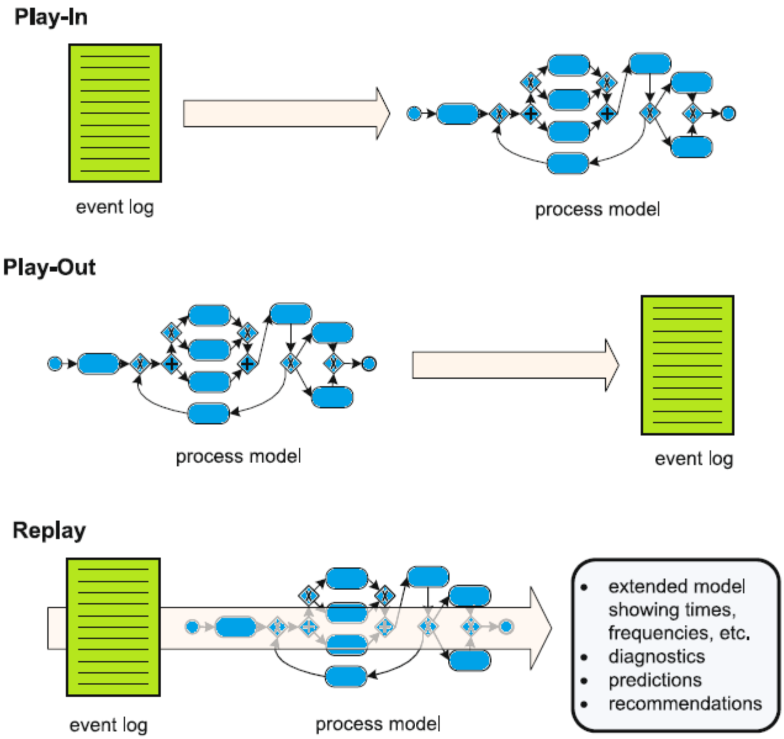


Figure 2.6: Three ways of relating event logs (or other sources of information containing example behaviour) and process models: Play-in, Play-out, and Replay. (Adapted from (80)).

h>, is replayed on the Petri net depicted in Figure 2.5, it will appear that d should have happened, although it did not.

- Extending the model with frequencies and temporal information: Replaying the log can help to point out which parts of the model are most frequently visited. In addition, the replay can also help to detect bottlenecks. Let us consider, for instance, the trace $<a^8, b^9, d^{20}, e^{21}, h^{21}>$, where the timestamps are denoted by superscripts. If this trace is replayed on top of Figure 2.5, we can see that *e* has been enabled at time 20 and has occurred at time 21. In this example, although *d* had already been enabled at time 8, it only occurred at time 20, and the time it took to complete it delayed the enabling of *e*. That is because the firing of *e* depends on the time that *d* complete first (in addition to *b*, which also finished before time 20).

- Constructing predictive models: Replaying event logs can help to construct predictive models by providing particular predictions for the different states of the model. For instance, if a predictive model is learned by replaying many cases on the Petri net depicted in Figure 2.5, it would show that the expected completion time after enabling e is eight hours.

- Operational support: Not only historic event data but also partial traces of running cases can be replayed, making replaying useful for detecting anomalies at run-time. For instance, the partial trace $<a^8, e^{11}>$, corresponding to a case that is still running and will never fit into Figure 2.5, in this case, will not allow it, and will generate an alert before the case completes.

2.3 Transition Systems

The most basic technique of process modelling is a transition system [80], which is one of the computation study concepts and consists of states and transitions. The transition happens between states; each one could be labelled with labels which are not unique and could appear on more than one transition. Transition systems differ from 'finite-state automata' in several ways, such as: i) it is necessary for the set of states to be finite or countable; ii) it is necessary for the set of transitions to be finite or countable, and iii) The state has no 'start' state or 'final' states.

Figure 2.7 represents the managing of a request for compensation at a business airline company. The seven states are represented as black circles, the single initial state being labelled 's1' and the final state labelled 's7'. There is a unique label for each state; each label is unique to identify the state. Transitions are represented as arcs, which establish a connection between two phases and are each labelled with the name of an activity. Multiple arcs can have the same label name. For instance, 'examine casually' appears twice in the transition system [88].

The transition system shown in Figure 2.7, can be formalised as follows: States (S) = s1, s2, s3, s4, s5, s6, s7, initial state = s1, final state = s7, Activities (A) = register request, examine thoroughly, examine casually, check ticket, decide, reinitiate request, reject request, pay compensation, and transitions (T) = (s1, register request, s2), (s2, examine casually, s3), (s2, examine thoroughly, s3), (s2, check ticket, s4), (s3, check ticket, s5), (s4, examine casually, s5), (s4, examine thoroughly, s5), (s5, decide, s6), (s6, reinitiate request, s2), (s6, pay compensation, s7), (s6, reject request, s7).

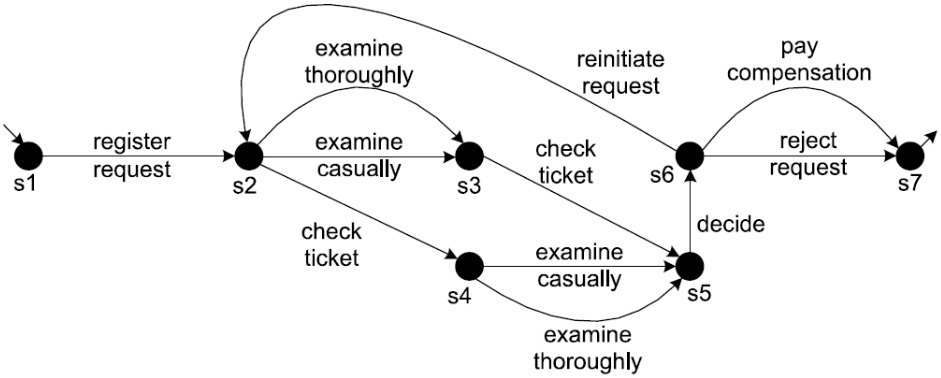


Figure 2.7: A transition system having one initial state and one final state. (Adapted from (80)).

The behaviour of a transition system can be explained as follows. The initial states determine the starting point for all of the possible transitions, in the sense that any path in the graph starting from such states corresponds to a possible execution sequence. In general, there are infinite possible execution sequences. In particular, a path is successful if starting from an initial state and one of the available final states is reached. On the other hand, a path deadlock occurs if a non-final state is reached without having any possible transition to be further performed. In addition, it is important to note that, in order for a path to be successfully ended, it is not enough to simply avoid deadlocks, but it is also necessary to avoid live-locks, where although further transitions are enabled, the final states cannot be reached [81, 80].

Due to the fact that any process model based on executable semantics can be mapped onto a transition system, many of the concepts developed within the field of such systems can be extended and translated into higher-level languages, such as Petri nets, Business Process Model and Notation (BPMN) and Unified Modelling Language (UML) activity diagrams. Let us address, for instance, the issue of determining whether two processes are the same from a behavioural point of view. In order to answer such a seemingly simple question, different equivalence notions defined for transition systems can be used, as suggested in [96]. In this line, the equivalence between both processes can be defined, on the one hand, based on the trace equivalence principle, which considers that two transition systems are equivalent if their execution sequences are the same. On the other hand, more refined equivalence notions, such

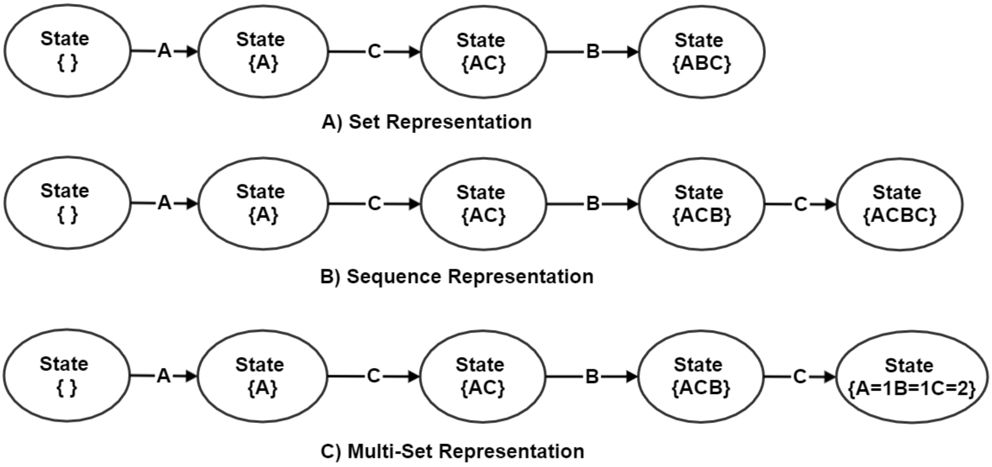


Figure 2.8: Transition System with (a) Sequence Representation, (b) Set Representation, and (b) Multi-set Representation of traces <ABBC>.

as the branching bi-similarity notion which suggests that the moment of choice should also be taken into account, can also be used. In this way, provided that the process models are expressed in a language with executable semantics, the equivalence notions initially defined for transition systems can be applied to any pair of models [80].

2.4 Preliminaries

This section describes the elements required for building our prediction model, which consists of two parts: firstly, to build an extended annotated transition system that includes a number of attributes that comprise relevant structural information about the traces; secondly, to apply regression techniques for predicting the remaining time of the process execution at each time.
Definition 6 (Partial trace, State) [88]: A partial trace PT is any part of a trace T that contains one or more events in sequence. For each (partial) trace, three state representations are defined [88]: Set, Multi-Set and Sequence. In this thesis, we focus on the Set representation as the basis for our model. In Set representation, each partial trace, PT, has associated a state, (PT), which is labelled through the activities in PT and where no repetition of activities is considered (no matter its order of execution).
Definition 7 (Transition System) [88]: A transition system (TS) is a triplet (S, PT, TR), where

S is the state space, PT is a set of partial traces, and TR is a transition relation which describes how the system moves from one state to another. The TS model has different forms depending on the state representation on which it is based.

In Figure 2.8 we show the differences between Sequence representation, Set representation and Multi-set representation in a TS, using a simple example of a process involving only three events with its corresponding activities (A, B, C). Here we consider the representation of trace <ACBC>involving the three activities A, B and C. In Figure 2.8a), we show the Set representation, where the final state is {ABC}, since Set representation does not consider the order of the activities within the trace. For example, other traces involving the same activities, like <CCBA>, or <BABC> will all have the same representation: state {ABC}. In Figure 2.8b, the final state of the traces simply is the sequence of the activities, which is (state {ACBC}). In Figure 2.8c), we show the Multi-Set representation, where the final state is {ABC} (as in the Set representation) where it does not consider the order of the activities within the trace, but it does consider the number of times each activity was executed.

Definition 8 (Annotated Transition System) [88]: An Annotated Transition System (ATS) is a two-fold system (TS, MF), where TS is a transition system based on the Set representation and MF is a measurement function $MF(S)$ that annotates each state S in the ATS. For instance, in [88], authors define MF as the time remaining since the occurrence of the last activity of each partial trace in the TS until that trace is completed.

Table 2.3: An example log, showing seven traces, each of them represented as a sequence of activities A, B, C, D, E, occurring in different orders. Superscript numbers indicate time-stamps at which each activity is completed.

#	Traces
1	$<A^{00}B^{06}C^{12}D^{18}>$
2	$<A^{10}C^{14}B^{26}D^{36}>$
3	$<A^{12}E^{22}D^{56}>$
4	$<A^{15}B^{19}C^{22}D^{28}>$
5	$<A^{18}B^{22}C^{26}D^{32}>$
6	$<A^{19}E^{28}D^{59}>$
7	$<A^{20}C^{25}B^{36}D^{44}>$

In Figure 2.9, we show an example of the ATS model defined in [88], built from the traces described in Table 2.3. Let us consider state {AB} and the partial traces it represents as the following:

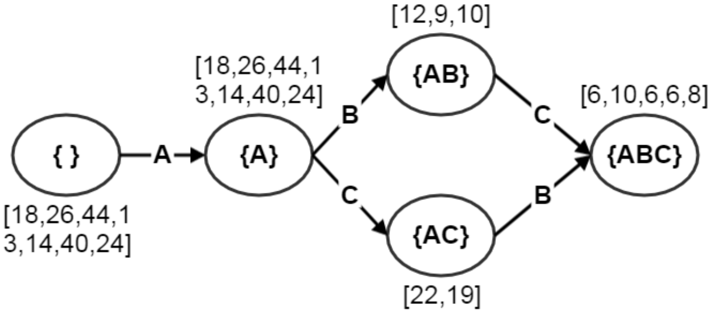


Figure 2.9: ATS-based on the log shown in Table 2.3

- $<A^{00}B^{06}>$ (from Trace 1).
- $<A^{15}B^{19}>$ (from Trace 4).
- $<A^{18}B^{22}>$ (from Trace 5).

The annotation of state {AB} ([12, 9, 10]) is related to the times elapsed from the timestamp of the last activity in each partial trace (6, 19, and 22 respectively) until the end of the corresponding trace (18, 28, and 32, respectively). Therefore, the respective differences between these values (12=18-6, 9=28-19, 10=32-22) form the annotation attached to the state {AB} [12, 9, 10].

2.5 Estimating of Remaining Time of Business Process

Let us recall again, there are three main types of process mining techniques [83]: process discovery, conformance checking, and process enhancement. Process discovery takes an event log and produces a model without using any *a priori* information [83, 80]. Conformance checking makes a comparison between a designed process model and the process discovered from the event log to show where the real process deviates from the designed one [83]. Process enhancement aims to extend or improve an existing process, using information related to the process which is usually extracted from the recorded event logs [79].

In process enhancement, temporal information is usually used to measure the wait times between process activities, to check the temporal behaviour during traces replay, to provide information about relevant issues in the process (e.g. bottlenecks, throughout times, frequencies) or to predict the remaining times from running process instances [83]. In this sense, predicting the remaining time of process instances (running cases) has been highlighted in the literature as one of the most important current challenges in process mining [82]. The remaining time of a process instance is the required time for it to be finished from a particular execution state. Accurately predicting remaining time is a key issue for all actors involved in business processes management. For organisations, having accurate time predictions allows them to optimally manage their resources [8], assess the quality of the services they provide as well as take appropriate managerial decisions in advance. For end users, it is also critical to be aware of when the processes they are involved in will finish [63, 71]. Some examples of the latter are bank customers applying for a loan, who need to know in advance how long it will take for their loan application to be reviewed, checked, assessed and accepted or declined, or in healthcare processes, where it is crucial to know the remaining time of each treatment in order to effectively manage the next treatments (and to prepare all the necessary resources in advance) or the next patients.

The problem with predicting the remaining time is part of a more general problem known as predictive monitoring. In recent years, several proposals focusing on predictive monitoring and, more specifically, on the prediction of remaining time have been presented [32, 53, 97]. Initially, these proposals have focused on the representation of the process executions or traces under the hypothesis that traces with different characteristics have different remaining times. Several of these approaches are based on Annotated Transition Systems (ATS), where each (partial) trace is associated with a state having a different representation [88, 85, 66, 63, 12]. Other approaches use a partial trace-based or index-based representation [8, 71, 98]. More recently, approaches have been proposed for applying machine learning methods used to predict the remaining time [95, 31, 78, 13, 76].

In Chapter 3, we will discuss in detail these related works, and we will also summarise their limitations with regard to time prediction accuracy.

CHAPTER 3

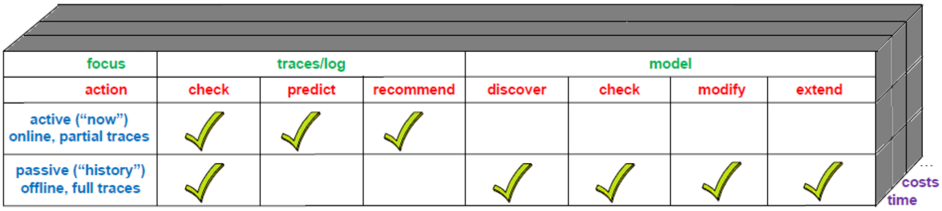
LITERATURE REVIEW

3.1 Theory Relevant to Remaining Time Prediction Using ATS

Within the revolution of predicting the remaining time of business processes, a number of proposals have been addressed. The models described in [88, 85, 66, 63, 12] are state-based representations known as Annotated Transition Systems (ATS). In this section, we are going to describe them in detail, since the model we will propose in this Thesis belongs to this category.

Before going into depth in the literature review, it is relevant to recall here that these models are centred around the Annotated Transition system concept (defined in Section 2.4). In Transition Systems, the process traces (real executions or instances) are represented as a sequence of states and a set of transitions between them: a state models a sequence of activities of the trace and a transition represents the execution of the next activity in that trace. Each state of a transition system is annotated with temporal information about the execution of the process, thus generating an Annotated Transition System (ATS).

In [85], a general framework for operational decision support based on the idea that process mining is not only limited to the past but can also be used for the present and the future has been proposed. In particular, a new set of time-based operational support approaches implemented through the process mining tool ProM has been introduced [90]. The proposed time-based operational support approaches are based on an Annotated Transition System that contains time information extracted from event logs. The ATS can be used to check (time) conformance with the time where cases are being executed [88], predict the remaining time of processes of incomplete cases and recommend convenient activities for the end users working



focus	traces/log			model			
action	check	predict	recommend	discover	check	modify	extend
active ("now") online, partial traces	✓	✓	✓				
passive ("history") offline, full traces	✓			✓	✓	✓	✓

Figure 3.1: The three major aspects of process mining are 1) Discovery, 2) Conformance, and 3) Enhancement. (Adapted from (85))

on these cases, before annotating each state with the average remaining time to complete, with each trace execution represented by this state. The average time is provided as the remaining time prediction.

In this way, the particular focus can be made on the active use of process mining involving partial traces corresponding to cases that have not been completed yet. Within the context of such running cases, three types of actions can be identified: check, predict and recommend. These actions, shown in Figure 3.1, are referred to as operational support, since they are aimed at influencing the process while it is running. In this way, the proposed time-based operational support approaches can be used to check the time conformance of running cases, predict the completion time of the cases and recommend appropriate activities to end users working on such cases towards minimising their overall flow time. The advantage of this work, in addition to the time prediction, is that it also uses two operational support techniques, which influence the processes during the running time. It focuses on the partial traces, and while the case is still running, *checks* the last task. For instance, this approach will check if this task fits the model or not, and use the *recommend* technique by recommending the next task based on the last task.

A weak point of [85] is that, in this work, authors only present results of predicted remaining time calculation but do not provide any comparison nor any error measurements. The paper describes a framework for operational support using process mining and presents how it can be used for predicting the remaining processing time, but lacks real validation.

In [63] a prediction model is proposed for forecasting the completion time of running cases (i.e. cases that are not yet completed), usually known as 'Time to completion', in the case of business processes. Within the context of the proposed approach, a service-oriented architecture providing a testbed for carrying out predictions on business processes is used. Several prediction techniques, such as descriptive statistics, regression and hidden Markov

Models (HMMs), are used in this work so that the predictions can be made based on event logs. In this model, the authors use the ATS introduced by Aalst et al. [88] for forecasting time to complete for business processes. The ATS is capable of abstracting event logs from past executions. In this work, the ATS approach is used together with sequence abstractions to predict the time to completion of business processes. Finally, a practical implementation of the system is proposed by simulating the execution of a real business process, and the obtained prediction results using the proposed prediction model are compared to the ones obtained using other state-of-the-art prediction models.

Systems based on path mining and ATS strongly depend on the existence (and availability) of historical event logs, since they assume that all the paths and patterns that are likely to appear during the execution of the process are contained in such logs. Nevertheless, this is not strictly true for business processes, where the paths and events are not equally likely, being process patterns, and therefore their occurrence likelihood is affected by several factors, such as resource availability and its characteristics. In such a dynamic process, where process patterns change over time, the steadiness assumption is not a suitable one. In this work, a prediction model, fitted by historical data, capable of being updated based on the systems time and content requirements, is proposed in order to handle the dynamics of the business process. In particular, to learn the workflows patterns available in historical event logs, a HMM is used. In this model, the authors built an architectural framework allowing the simulation of business processes and prediction techniques. They also proposed a HMM-based prediction model for predicting the time to completion of business processes. Finally, they built a prototype test-bed that implements (via simulation) the proposed architecture.

The most important limitation of this approach is that the validation results they present only consider a single synthetic dataset with 14 tasks. They do not provide error results, but a simple qualitative comparison between the predicted remaining time values given by each technique and the baseline work [88], showing that the similarity of slope values for ATS and one of their techniques validate the implementation of the simulation system.

In [88], a configurable set of abstractions providing a fair balance between over and under-fitting is used to obtain reliable time predictions. In addition, the proposed approach is practically implemented in ProM. The prediction approach proposed in this work is an improved version of the prediction service presented in [95]. Including a completely new approach for time prediction in order to overcome the limitations encountered in the one proposed in [95]. In particular, unlike other existing approaches in the literature, where the problem is reduced to a ‘simple heuristic’ process, such as using a regression model and estimating half of the average time or the average time minus the already elapsed time, an ATS capable of representing an abstraction of the process with time annotations was used in this work.

In [95], an information system based on recording any event that takes place, such as, for instance, the commencement of an activity and the building of the process model based on the recorded information, is proposed to support operational processes. In this work, a transition system is developed using a set of different abstractions, as proposed in [87]. In particular, the transition system is annotated with information about elapsed, sojourn and remaining times, accounting for the average time to reach a particular state, the average time spent in the state and the average time to reach the end of this state respectively. Based on this annotated information, the transition system is able to predict the remaining time of all (or some) of the running cases, namely of the process instances. In particular, within the context of this approach, the process instances strongly depend on the transition system generation as well as its corresponding abstractions presented in [87]. In this way, the transition system would be able to keep the balance between over- and under-fitting with respect to the log, thus providing better predictions. The advantage of this work is that it provides a base framework for the ATS-based model, where subsequent works rely on their base model.

The limitation of this work is that the results showed low accuracy in the prediction of the remaining time. Whereas in the results of the first event log case (the WMO process of a municipality), the remaining time prediction is about 64 days (MAE), the results of the second case study (the WOZ process of a municipality) is 17 days (MAE). This occurs, as we explained before, due to the low number of information annotated in the ATS model, this model only annotating temporal information and taking the prediction of the remaining time as an average for all of the historical remaining times. Moreover, the approach assumes that the processes are in a steady state. In many cases, this is not realistic since the model may change over time. In addition, information related to the context of a case is also important when predicting flow times. Examples of this information are: the other cases it is competing with may place the next cases in a queue, the availability of resources and some external factors influencing the process such as the weather, the time of the day, traffic conditions, etc.

Ceci et. al. [12] proposed a new approach for dealing with operational support in process mining, focusing especially on the prediction of the next activity as well as the time to complete. In particular, the proposed approach intends to address some of the most common issues within the context of the operational support, such as incompleteness, robustness to noise and over-fitting. Along this line, the proposed approach is aimed at identifying partial process models to train both types of predictive models, in other words, the model for predicting the next activity and the model for estimating the time to complete. In order to do so, a two-step approach based on data mining techniques, as suggested in [11], where a hybrid data mining approach is used to deal with the associative classification task, is carried out. In the first step, descriptive data mining, specifically a tailored sequential pattern mining algorithm, is used for partial model mining. In this way, prediction models that are robust against incompleteness and that are not over-fitted can be obtained. In the second step, predictive data mining is used to mine nested classification/regression models. Here, it is important to highlight that the use of nested models is selected in particular since it allows the application of any traditional classification or regression technique.

As discussed above, the proposed approach is aimed at identifying frequent partial processes. In order to do so, an efficient frequent pattern mining algorithm is used to extract frequent activity sequences that are then represented as sequence trees. Then, each node of the resulting tree is associated with a particular prediction model, taking into account not only traditional attributes, such as the performer of each activity, but also additional attributes, such as the cost associated with the event or the location where the event is taking place. This kind of prediction model is called a nested model. The described implementation of the proposed approach allows for handling incompleteness, robustness to noise and over-fitting by removing unusual behaviours through the sequence mining algorithm. In addition, the proposed nested models allow certain flexibility, in the sense of allowing the plugging-in of any classification/regression learning algorithm, as well as enabling different representations of the data (one for each node of the tree).

To sum up, the authors propose here a prediction model for business processes based on: (i) a partial process model's discovery through sequential pattern mining techniques, as well as; (ii) the inclusion of additional useful information regarding activities associated with particular partial process models in order to train nested prediction models on event logs. Specifically, the prediction models are focused on predicting the next activity and the time to completion of a new (running) process instance.

A weak point of this work is that there is low accuracy in the prediction of the remaining time due to the limit of the attributes in the model used, as they only consider the sequence of events. Regarding the result provided, the prediction accuracy is improved by approximately 30% with ProM Data and 11% with THINK3 Data. The authors also validate their work using the non-noisy dataset, which in general will produce better results than if they had used noisy datasets, typically with even lower accuracy.

According to Polato et al. [66], in order to successfully deal with processes under service-level agreement constraints, it is crucial to be able to accurately predict the time to completion of the business process instances. Nevertheless, there are several factors that can impact the process instance tendency, making it hard to achieve the required accuracy by simply resorting to the time statistics of historical cases. In order to improve the prediction accuracy, a new approach combining the control and the data flow perspectives is proposed in this work. Along this line, the process model is enriched by adding the relevant time and data information with a view to predicting the time to complete. In particular, the time to complete the prediction of a running case is computed, resorting to: (a) the likelihood of all the following activities, given the already-collected data, and (b) the remaining time estimation based on a regression model built on the data.

This work proposes a new technique that can be used in operational settings for predicting the time to complete a business process. In particular, the proposed approach is based on multiple perspectives, in the sense that both the flow of the activities of the running case, as well as the data that the current process instance is generating, are considered for performing the prediction. In particular, from the former perspective, i.e. in terms of the control-flow, the information is encoded by a transition system, while from the latter perspective, all of the data recorded by each activity is collected and used to refine the prediction.

In general, approaches that are already available in the literature use the transition system to store information associated with a specific trace. In this work, the transition system is annotated with three additional entities. The first accounts for the average time spent on every state. The second consists of a Naïve Bayes classifier, associated with every state, capable of determining, given the set of data attributes, the probability distribution over the set of reachable states from the current one. The third is a Support Vector Regressor (SVR) that, given the set of data attributes, can predict the time to complete for each transition. Figure 3.2 shows a typical application scenario of the proposed approach. According to the process of a bank application process, depicted in Figure 3.2 the proposed approach flows in the following way: given a partial trace containing the log of various already-executed activities, all of the data attributes observed until that moment can be collected. In particular, after activities A, B and C have been executed, data attributes like the **Amount** of $1,000, the **Customer category** of gold and **Payments required** of 10, are available. Then, based on the available information in terms of the history of the trace and the set of data attributes, the time to complete the running instance can be predicted through the transition system annotated with the three additional entities previously described, namely the entity accounting for the average time spent at each state, the entity based on the Naïve Bayes classifier and the entity based on the Support Vector Regression.

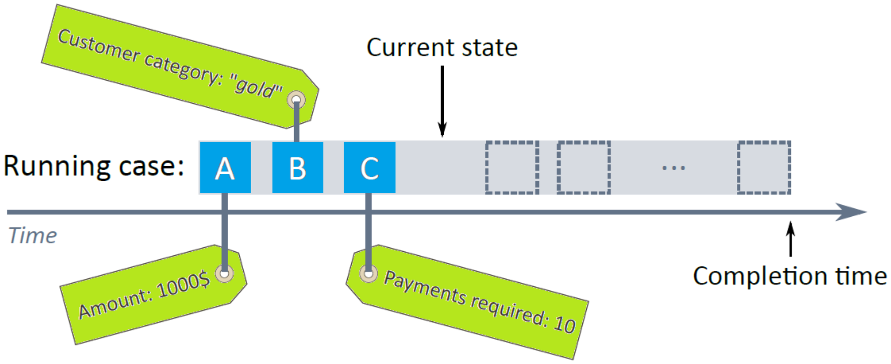


Figure 3.2: A representation of an application scenario. (Adapted from (66))

The weak point of this work is that it has low accuracy in the prediction of the remaining time due to the limit of the attributes in the model used. Only two attributes (Sojourn time and Remaining time) are annotated, which do not provide enough information to let the model learn in the best way. Regarding the result provided, the authors evaluate the model with a log with 1,500 traces, and the improvement in MAPE with the base-model [82] is about 15.5%, whilst for a log with 5,000 traces, it is 4.75%. The improvement decreases as the number of traces increases.

3.2 Theory Relevant to Remaining time prediction using non-ATS model

In this section, we will describe other related works that do not follow the ATS-based approach, and which are machine learning methods to predict the remaining time. Nowadays, the capability of predicting a process outcome in order to be able to make a priori recommendations regarding which is the best way to move forward from a certain point in the process has become a crucial task. In particular, a priori recommendations can be supported by different kinds of predictions, such as regarding the process' remaining time or regarding the process cost.

In [8], a novel prediction approach for the process' remaining time based on query catalogues is proposed. Query catalogues are groups of partial trace tails from all of the traces available in a log. A partial trace tail is annotated with the number of its occurrences and the sum of its remaining times. Then, the prediction of the remaining time to completion for each partial trace is the average time in the catalogue where the trace is. Query catalogues incorporate useful elements from annotated transitions systems [88], so that a collection of annotated partial trace tails can be built with these elements. In this work, the information of process events is stored in the form of partial trace tails by the query catalogues in order to use them to estimate the remaining time of new executions of the process. In addition, query catalogues are created considering all of the possible combinations of events for all the traces of an event log. In this way, although more memory is used to compute the process' remaining time, the needed processing capacity to do so is reduced. Three different new methods for calculating the remaining time for partial traces based on query catalogues are proposed in this work. These approaches not only allow for improving the estimation based on the new information being collected compared to the baseline work of [88], but also make the prediction system more flexible and dynamic, For instance, the obsolescence of certain partial trace tails or the integration of new ones could be determined by simply deleting the obsolete information or adding the new one to the catalogues respectively. In such cases, the proposed approach avoids the need for reprocessing all of the previous data, thus reducing the computational cost.

The weak point of this work is the modest accuracy value of the predicted remaining time for partial traces. Three different new methods are used for calculating the remaining time for partial traces, but all of them are based on calculating the average of the historical records. Regarding the result provided, using a real event log of Chilean telecommunications compared with the baseline work of [88], the best method (Average catalog) improved about 1.5 hours in MAE, and one hour in RMSE, and about 50% in MAPE as shown in Table 3.1.

Rogge-Solti et al. [71] mention that in order to meet specific deadlines or service-level agreements, companies usually need to speed up the execution of their processes. In general, this is done either by raising alerts or by using additional resources. In such a context, it becomes crucial to accurately predict the remaining time of a case as well as the risk of missing a deadline. In order to do so, a particular kind of stochastic Petri Nets, which can identify ar-

Table 3.1: Comparison results of different methods described in (8) applied to a real event log.

Method	MAE (days)	MAPE (%)	RMSE (days)
Simple heuristic (SH)	19.04	**104.56%***	20.19
Annotated transition system	17.09	195.52%	19.06
Default horizon and catalog (DRT)	15.76	148.24%	**18.02**
Average catalog (RT)	**15.57**	146.83%	**18.02**
Best horizon and catalog (RRT)	15.83	154.05%	18.43

bitrary duration distributions, can be used. In this work, the temporal performance of business processes is studied. In particular, expressive probabilistic models which allow the possibility of including information extracted from event data [70] are used. In this way, by including the information of the elapsed time since the last event, the accuracy in the prediction of the remaining duration [88, 27, 70] as well as the risks of missing temporal deadlines or guarantees [19, 64] can be increased. In addition, reasonable wait time guarantees, such as a wait time that is met in 99% of cases, can also be computed. Finally, the prediction approach proposed in this work can be used within the context of resource management, where the scheduling success strongly depends on the accurate prediction of the remaining time of the activities [22].

To sum up, a new approach for predicting the remaining time as well as for estimating the chance of missing a given deadline within the context of business processes is proposed in this work. However, the implementation of the proposed prediction approach depends on some assumptions. In particular, due to the fact that the information of the elapsed time since the last event is included in the prediction model, the effects of time are more significant in the presence of long time spans between observable events in the business process. The limitation of this work is that it depends on the elapsed time only and that the prediction accuracy when the duration times are short is lower than with long durations. Based on the result provided from the comparison with the baseline work [88] in two real-life event logs, the result of the Business Process Intelligence Challenge 2012 log improved the RMSE average by about 2.5 days, and the result of the shipment import process improved the RMSE average by about 2 days. This model provided very low improvement regarding the baseline work [88].

In [98], a white-box flow analysis approach is proposed to predict quantitative performance indicators of running process instances, especially focusing on predicting the remaining cycle time of such instances. The main idea of this work is to use flow analysis techniques to include the performance indicator predicted at the activity level into the process instance level. In general, researchers in the field resort to black-box approaches, either based on stochastic or regression models, to predict the remaining execution time of a process instance. In such cases, the prediction result is expressed as a single scalar value, and no explanation is given in terms of elementary components. On the other hand, quantitative performance indicators, such as cost and time, are aggregations of the corresponding performance indicators of the process activities.

In particular, the cycle time of a process instance consists of the sum of the cycle time of the activities performed in such a process instance. Although there exist different techniques in the literature that can predict the aggregate value of a performance indicator for a running process instance, to the best of the author's knowledge, none are able to explain to what extent each activity contributes to the aggregate prediction. In order to address such an issue, a white-box approach to predict quantitative performance indicators of running process instances based on the well-known flow analysis for quantitative process analysis is proposed in this work. Flow analysis techniques take into account the control-flow relations between the process activities to estimate quantitative performance indicators at the level of the process by aggregating their estimated values at the level of the activities. Similarly, in order to predict the remaining cycle time of a process instance, the cycle time of each activity that might potentially be executed within this process instance is first estimated to be then aggregated using flow analysis.

The limitation of this work appears when the model facing the cycle calling for the same event or another event, which is multiple occurrences of the same fragment of activities in a row, leads to the limitation of flow analysis-based approaches. Regarding the result, the evaluation has been done depending on the different Prefix[1] length, where the logical comparison should have also all prefixes.

In [95], the prediction of the remaining execution time based on event logs is addressed. In particular, special attention is paid to explain what the remaining execution time of a specific partial case is, i.e., accurately answering the customer's request regarding the time needed to handle their claim. Unlike many other previous prediction methods proposed in the literature which used parametric regression techniques, non-parametric regression techniques are

[1] Prefix log is a custom log that includes n number of activities in each trace of the original log. If $n = 20$, then the length of each trace in the prefix log is 20 or less if the trace length is lower. [97]

used in this work since they have been shown to be better suited for dealing with unspecified business processes [65]. In these cases, little or no precedents are available, making it impossible to assume any specific distribution for the execution time. It is then necessary to resort to non-parametric regression approaches which, contrary to parametric approaches, do not need to know the form of the relationship between the predictor variables and the target variables, it being enough to assume that there is some (although unknown) kind of relationship between them. In particular, the so-called smoothing or 'local averaging' non-parametric regression is used to make estimations based only on the observed data, without the need for a parameterised model.

Several methods for estimating the remaining execution time are introduced in this work. On the one hand, a naïve approach based on the average execution time over a log is proposed. On the other hand, regression techniques based on kernel functions, based on the ones introduced in [68], capable of performing non-parametric regression with both continuous and un-ordered categorical variables, are also proposed. In particular, three different types of regression models, based on the occurrences of activities, the duration of activities, and the case data, which corresponds to ordered ordinal, continuous and un-ordered ordinal variables respectively, are presented. The three proposed prediction models have the same structure, consisting of a set of measurements, a kernel function and a predictor. In order to actually implement them, it is necessary to perform regression on a combination of different variable types, namely the variables based on the occurrences of activities, those based on the duration of activities and those based on the case data. Each of these variable types is represented by measurement and target variables.

In particular, the same target variable is defined for all the measured variables. In addition, all the measurement sets are the same size, in the sense that one measurement is taken per non-empty prefix of a case in the log. In this way, all measurements can be easily combined into a single vector containing the variables based on activity occurrences, activity duration and the case data for each prefix.

The weak point of this work is that validation consists of comparing the four different proposed methods with the 'average estimator' (a naïve approach using only the average cycle time over a log). A single real dataset is used ('bezwaar WOZ', from a Dutch municipality) here. Results show that three of the approaches outperform the naïve approach. The reported Mean Square Error for the best method is 1,532.85.

In [31], a novel framework for predictive monitoring business processes towards estimating the actual probability of fulfilling a given predicate upon its completion within a running case is proposed. In order to do so, the authors of this work propose to consider the sequence of events observed in the current trace as well as the data attributes associated with such events. The proposed prediction approach is carried out as follows: on the one hand, control-flow information is used to cluster the traces of the previous, already completed, cases without considering the event payloads (data attributes). Then, a classifier, based on event data attributes, is built for each of the previously defined clusters in order to identify the cases in which the predicate being currently considered is likely to be fulfilled and separate them from the ones leading to a violation within the cluster. Finally, at run-time, the prediction of a particular running case is performed by assigning it to a cluster and applying the corresponding classifier. In order to practically implement the proposed prediction approach, three methods should be selected, namely one for encoding traces in the event log as feature vectors, one for clustering and one for classifying. The widely used ProM process mining tool [90] has been used to perform the experiments, placing special emphasis on applications of Operational Support (OS) [85, 101, 49]. Within the OS environment, a stream of events, such as the ones produced by an enterprise system, are considered to be inputs, while a set of predictions for each new incoming event are used as updates.

Finally, the predictive monitoring approach based on historical control flow data as well as data attributes proposed in this work has the advantage of significantly reducing the run-time overhead while maintaining competitive accuracy. The key aspect of reducing the run-time consists of the fact that the above-mentioned classifiers are built offline. In this way, the online phase, i.e., the run-time step, simply consists of matching an uncompleted trace of the running case to a cluster so that the corresponding classifier can be applied and the probability of fulfilment of the monitored predicate can be estimated.

The weak point of this work is that it does not provide any error measure metric. Although in the results section the authors provide the temporal information they predicted, this information does not allow us to figure out the prediction accuracy of the proposed model.

Niek et al. [78] mention that most of the available methods in the field of business process monitoring are tailor-made for specific prediction tasks and are not easily generalisable. Moreover, even their accuracy is strongly dependent on the data set being used as well as on the point in time at which the prediction is made. For instance, a particular technique may outperform another one for a given log and prediction point, but under-perform for another log at the same prediction point or for the same log at an earlier prediction point [31, 56]. In such a context, researchers are usually compelled to combine multiple tuning techniques [56], resort to trial-and-error or to apply considerable tuning, such as hyper-parameter optimisation [30], in order to achieve the required consistency in the obtained accuracy.

There have also been recent prediction models which use Deep Learning Neural approaches, such as Recurrent Neural Networks (RNN), with LSTM architectures [40] aiming to provide accurate predictions in this field, in the same way as they have done in other fields such as, for example, computer vision, natural language processing [57] or speech recognition [35]. Namely, in Evermann et al. [25], LSTMs are used within the context of predictive process monitoring, in particular, to predict the next activity in a case.

The following research questions intend to be answered: (i) what is the actual possibility of using LSTMs in a wide range of predictive process monitoring applications?; (ii) how can they be implemented in such contexts? and; (iii) can LSTMs achieve high accuracy across different prediction tasks, event logs and prediction points? In order to address these questions, in this work, the LSTM models are used to predict: (1) the next activity in a running case and its timestamp, (2) the continuation of a case up to completion and (3) the remaining cycle time. Experiments comparing the accuracy of the proposed LSTM-based prediction model with other state-of-the-art, tailor-made methods at different prediction points, using three real-life event logs, have been carried out.

The limitation of this work is that it performs worse than the baselines [88] when the trace includes a lot of event cycles, as shown in Figure 3.3 for b) Business Processing Intelligence 2012 dataset - work item (BPIC12w) log. However, in c) BPIC12w log with no repeated event, the result was more enhanced than the one with no repeated event. The result clarifies that LSTM model cannot deal with such a log as BPIC12w and for other BPIC real-life logs where they have the processes with many repetitions [3].

Cesario et al. [13], proposes a framework for evaluating and predicting the performances of business processes, based on historical data. The proposed approach is intended to predict the run-time of different performance measures, such as the remaining processing time/steps for uncompleted process instances, using inductive-learning techniques to build a modular representation of the process and modelling the most relevant process variants (in terms of performance) by different regression models in order to discriminate them based on context variables. The proposed prediction technique combines different data mining methods, such as non-parametric regression methods, probabilistic trace clustering schemes, and a novel tailor-made data transformation mechanism, towards achieving a suitable level of abstraction

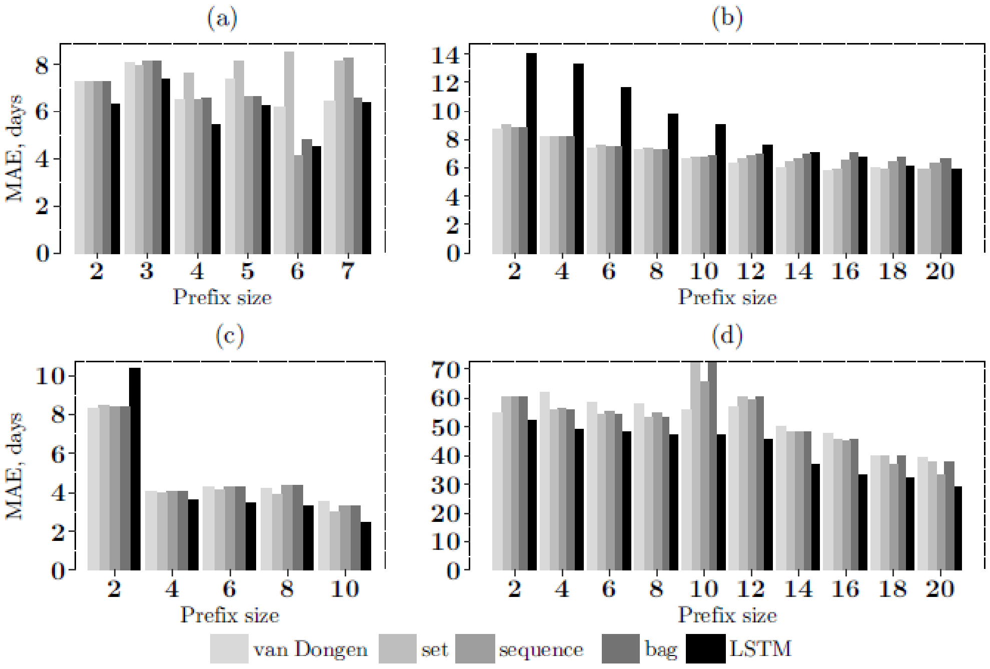


Figure 3.3: MAE values using prefixes of different lengths for helpdesk (a), BPIC12 W (b), BPIC12 W (no duplicates) (c) and environmental permit (d) datasets.

for the logs. In addition, to bridge the gap in the literature regarding scalability issues and to be able to actually handle large logs, the proposed approach implements the computation of the trace clusters as well as their predictors in a parallel and distributed manner on top of a cloud-based service-oriented infrastructure.

The main contribution of this work is the combination of performance prediction and clustering techniques [28, 6]. In this way, the proposed approach intends to provide solutions for the main issues in the field. On the one hand, a quick log sketch is performed towards clustering the log traces by picking up a fixed number of performance values at predefined positions within the traces that, although rougher than the one introduced in [28, 6], still allows the user to recognise groups of traces with similar performances over the time, as the obtained experimental results demonstrate.

In order to increase the accuracy of the obtained cluster predictors, usually degraded by the underlying approximated representation of the logs and the use of greedy clustering algorithms, the traditional prediction clustering approach based on logic presented in [7] and used in [28, 6] is replaced with a probabilistic clustering scheme. Finally, as already mentioned, the scalability limitations encountered in [28, 6] are addressed by making the proposed approach suitable for large logs by computing probability-aware trace clusters as well as predictor clusters in a parallel and distributed manner based on the grid services approaches introduced in [14] for Distributed Data Mining (DDM) tasks.

In particular, the grid services have been developed according to the Web Services Resource Framework (WSRF) specifications of the WS-Core (Globus Toolkit 4 (GT4) [29]) and have been deployed onto a private cloud-computing platform. Here, it is important to highlight that resorting to a cloud infrastructure [44] to automatically deploy virtual machines hosting a GT4 container is highly recommended in the literature (see, for instance, [58, 75]) since it provides the authors with a flexible and customised environment for transparently and efficiently running their prediction approaches.

The limitation of this work is that it used two abstraction modes (multi-set and set) to summarise any trace in the log into a vector, which represents the behaviour of the trace by the occurrence of events only. Therefore, this information leads to low accuracy when predicting the remaining time, and more attributes are needed to enrich the model. The result provided shows that this model improved the time prediction by about 0.015 days in the MAE and 0.008% in MAPE.

After reviewing all of the models in the literature, the prediction of the remaining time of all models mentioned in this chapter has shown a few improvements compared to the baseline, some of them showing a comparison with other models and others providing only their own results. Niek et al. [78] present a model with good results compared to the baseline, not in the early stage of the running case, but later with more time consumed.

3.3 Conclusions

In addition to the above, in all of the previous revolutionary approaches, the main problem is that the encoding includes information about the context of the process execution state, such as the duration of the activities or about domain variables. The main problem with all these approaches is that their trace representation (or encoding) does not include all the relevant information related to the traces' execution, such as n-size loops between activities, the distance between activities or co-occurrences. Without this information about the structural features of the traces, it is difficult to make accurate predictions about the remaining time. Therefore, the need for accurate remaining time prediction models has become necessary.

In the next Chapter 4, we introduce a new remaining time prediction model, an extended ATS-based approach that considers structural features or attributes related to process execution. In our approach, each ATS state is annotated with vectors that contain information related to trace structures.

CHAPTER 4

A NOVEL TIME PREDICTION MODEL BASED ON STRUCTURAL INFORMATION FROM THE TRACES

4.1 Introduction

After revising the most relevant remaining time prediction models in the literature. In this chapter, we describe our proposal of a model that considers structural information of the traces to predict the remaining times in a business process.

Our approach consists of: i) defining a number of attributes on the business logs that capture structural information from the traces, ii) extending the well-known annotated transition system model to annotate its states with the values of the attributes and iii) applying linear regression for predicting the remaining time of the process for each state using the attributes values.

Information systems managing massive and variant business processes usually store all the transaction data into the form of event logs [95]. In business process management, the use of event logs is not limited to storing the data generated by the business processes. This historical information can be used to build predictive models that can be used to let running process instances learn from the previous records. By extracting the timestamp from the event log and extending it in the process model, it is possible, for instance, to measure the wait times between the process' activities. These can be used to let running process instances learn from the historical records.

One of the most imperative difficulties is predicting the remaining processing time of running cases [8, 20, 82, 66], defined as the required time for a process to be complete. Its accurate estimation during a process run-time is an issue that has been raised recently as one of the challenges in business processes enhancement [82].

In this chapter, we present a new vector-based and ATS-based approach that considers structural features or attributes related to the process execution such as frequencies, repetitions and cycles. Each ATS state is characterised by a set of vectors whose components are the values of these attributes for each partial trace that fits this state. Based on these vectors and on the remaining times of the traces related to them, a linear regression-based predictor has been built for each state.

In this thesis, we used the ATS-based model as the base of our work, therefore, in this chapter, we focus mainly on the comparison with Van der Aalst approach in [88], as it is the base work of other approaches in the literature. In addition, our ATS-based model will provide the necessary model for conducting the experimental segment.

4.2 A new prediction model for remaining time estimation

In this section, firstly, we will present the most relevant elements of an ATS [88], since our model proposes an extension of ATS. These elements were defined in the preliminaries in Section 2.4. Using these definitions (Event, Trace, Event Log, Partial Trace, State, Transition System and ATS), we will propose what follows an extension of the ATS model that includes a set of new structural features which are extracted from traces and capture relevant information about the traces which have an impact on the remaining time estimations.

In Figure 4.1, we show how an ATS is built, using the two traces shown in Table 4.1:

Table 4.1: Two examples of traces. The superscript in each activity indicates its timestamp (ending time).

Trace no.	Trace	Activities involved
1	$<A^3A^{18}B^{24}A^{30}>$	A, B
2	$<A^7B^{10}B^{15}C^{22}C^{31}>$	A, B, C

Figure 4.1 shows how the ATS model was built from the sample log that includes two traces in Table 4.1, where it shows each state and the partial traces related to it.

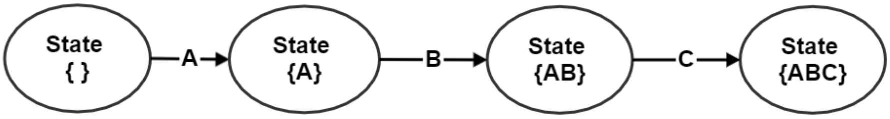


Figure 4.1: ATS model of the two traces shown in Table 4.1

For each state we have one or more partial traces as the following:

- State {A}:
  - T1: has two partial traces <*A*>, and <*AA*>
  - T2: has one partial trace <*A*>
- State {AB}:
  - T1: has two partial traces <*AAB*>, and <*AABA*>
  - T2: has two partial traces <*AB*>, and <*ABB*>
- State {ABC}:
  - T2: has two partial traces <*AABC*>, and <*AABCC*>

4.3 Trace Features

We extend the ATS model by considering a number of features (or attributes) extracted from the analysis of the event log traces. Each of these features is related to a measurement with which the ATS model will be extended, that is, each state of the ATS model will be annotated with both a set of attributes and the remaining time. A key difference between our approach and others in the literature is that the attributes we consider provide specific structural information about traces, such as the occurrence of the activities, its elapsed time or the existence of loops, among others. This structural information will act as predictor variables that will be taken into account by a regression model, aiming to improve the accuracy in the calculation of the remaining time prediction in a running process [2]. To consider the trace feature, We will define the following eight attributes:

Definition 6 (Occurrence, *Occ*): Let *PT* be a partial trace. We define $Occ(A_i,PT)$ as the number of times activity A_i occurs in *PT*. For example, for partial trace *PT*=<CCABBCAA>, we have $Occ(A,PT)$=3, $Occ(B,PT)$=2, and $Occ(C,PT)$=3.

Definition 7 (Cycle, *Cyc*): Let *PT* be a partial trace, and *LargSeq*(A_i) the largest sequence of activity A_i in *PT*. We define *Cyc*(A_i,*PT*):= (*length*(*LargSeq*(A_i))-1) as the number of times the activity A_i is repeated in sequence in *PT*. For example, for partial trace *PT*=<CCCABCCAA>, we have *Cyc*(*A*,*PT*)=1, *Cyc*(*B*,*PT*)=0, and *Cyc*(*C*,*PT*)=2.

Definition 8 (Position, *Pos*): Let *PT* be a partial trace *PT*, and *Pos*(A_i) the index set of activity A_i in that partial trace *PT*. We define *Pos*(A_i, *PT*):= maximum(*Pos*(A_i)) as the last happening of A_i in *PT*. It represents the last index of an activity happened in the partial trace. For example, for the partial trace *PT*= <CCCABCCAA>, we have *Pos*(*A*,*PT*)=9, *Pos*(*B*,*PT*)=5, and *Pos*(*C*,*PT*)=7.

Definition 9 (Distance, *Dis*): Let A_i be an activity at index *i* in a partial trace *PT*. *Dis*(A_i, *PT*) is the distance between the last occurrence of A_i and the previous one (backwards) in *PT*. *Dis*(A_i, *PT*)=0 in case A_i is a single activity in *PT*. For example, for the partial trace *PT*=<CCC ABCCA>, we have *Dis*(*A*)=3, *Dis*(*B*)=0, and *Dis*(*C*)=0.

Definition 10 (Duple, *Dup*): Let *PT* be a partial trace and state *S* the state associated to *PT*. For all the pairs of activities A_i and A_j in *S*, we define *Dup*(A_i,A_j,*PT*) as the number of times that the sequence A_iA_j happens in *PT*. For example, for the partial trace *PT*=<*CACACCAB*>, we have *Dup*(*A*,*A*,*PT*)=0, *Dup*(*A*,*B*,*PT*)=1, *Dup*(*A*,*C*,*PT*)=2, *Dup*(*B*,*A*,*PT*)=0, *Dup*(*B*,*B*,*PT*) =0, *Dup*(*B*,*C*,*PT*)=0, *Dup*(*C*,*A*,*PT*)=3, *Dup*(*C*,*B*,*PT*)=0 and *Dup*(*C*,*C*,*PT*)=1.

Definition 11 (Change): Let *PT* be a partial trace and *LA* its last activity. *Change*(*PT*), defined as the number of times the activities move from one activity to another from the beginning of *PT* until *LA*. For example, for the partial trace *PT*=<CCCABCCA>, we have *Change*(*PT*)=4, since it represents the move from activity *C* to *A* (first change), then the move from *A* to *B* (second change), then the move from *B* to *C* (third change), and finally the move from *C* to *A* (fourth change).

Definition 12 (Single): Let *PT* be a partial trace. We define *Single*(*PT*) as the number of single activities that have no more than one occurrence from the beginning of *PT*. For example, for a partial trace *PT*=<CCCABCCA>, we have *Single*(*PT*)=1, this partial trace has only one single activity (*B*).

Definition 13 (Elapsed Time, *Elt*): Let *LA* be the last activity in a partial trace *PT*. We define *Elt*(*PT*) as the time passed since the beginning of *PT* until *LA*. For example, taking the *PT* of the second trace in Table 2.3, <$A^{10}C^{14}B^{26}D^{36}$>, we have *Elt*(*PT*)=36.

All the attributes we consider are related to structural features that are of interest for characterising traces and/or partial traces. For instance, the occurrence of activity is related to the repetitions in traces; cycle and duple are related to the existence of loops in traces; whilst changes of events and their position are related to variety in traces.

Table 4.2: Value of the attributes for the partial trace PT, where superscripts of the activities indicate their timestamps. PT =<$A^2B^7B^{13}B^{22}A^{30}A^{37}B^{50}B^{54}A^{60}B^{62}$>

Attribute	Description	Representation	Attribute Value
Occ(*A*,*PT*)	How many times A occurs.	ABBBAABBAB	4
Occ(*B*,*PT*)	How many times B occurs.	ABBBAABBAB	6
Cyc(*A*,*PT*)	The maximum continuous repeat of A	ABBBAABBAB	1
Cyc(*B*,*PT*)	The maximum continuous repeat of B	ABBBAABBAB	2
Pos(*A*,*PT*)	The position of last occurrence of A	ABBBAABBAB	9
Pos(*B*,*PT*)	The position of last occurrence of B	ABBBAABBAB	10
Dis(*A*,*PT*)	The distance between the last occurrence of A and the previous one	ABBBAABBAB	2
Dis(*B*,*PT*)	The distance between the last occurrence of B and the previous one	ABBBAABBAB	1
Dup(*A*,*A*,*PT*)	Duple of AA	ABBBAABBAB	1
Dup(*A*,*B*,*PT*)	Duple of AB	ABBBAABBAB	3
Dup(*B*,*A*,*PT*)	Duple of BA	ABBBAABBAB	2
Dup(*B*,*B*,*PT*)	Duple of BB	ABBBAABBAB	3
Change(*PT*)	Change of activities	ABBBAABBAB	5
Single(*PT*)	Number of single activities	ABBBAABBAB	0
Elt(*PT*)	Timestamp of the last activity, as annotated in the caption	ABBBAABBAB	62

In Table 4.2, we present an illustrative example of the calculation of the previously defined attributes for the example trace <ABBBAABBAB>. We can see in this example that some of these attributes (e.g., *Change*, *Single*, *Elt*) produce a single value from each trace. Other attributes (*Occ*, *Cyc*, *Pos*, *Dis*) produce a number of different values which is in linear order with the number of events, N, in the trace. Therefore, In the example, N=2, we have two values for each of the attributes, so eight in total. Finally, the last attribute (*Duple*) produces N^2 values (four in the example, for all of the combination pairs of the activities in the trace, A and B).

4.3.1 Extended Annotated Transition System

Previously, we introduced the definition of the ATS model, as well as the definitions of the attributes we consider to be extracted from each trace in order to build our model. In this section, we will integrate these attributes into an Extended Annotated Transition System (EATS), and again we will define the List element in order to clarify more details about how we build the model.

Definition 14 (List): Let S be a state of an annotated transition system ATS associated with a given set of Partial Trace PT_j, $j = 1,...,P$ (P being the total number of Partial Traces associated with S). Let $\{Att_i, i = 1,2,...,N\}$ be the attributes defined in Section 4.3, and element E_j=$[valueOfAtt_1,...,valueOfAtt_M,RT_j]$, $j = 1,...,P$ a vector made up of the attribute values for each PT_j, being M the total number of attribute values and RT_j the remaining time for PT_j (time until trace completion, as defined in Section 4.3). We define $List(S) := \{E_1,E_2,...,E_P\}$ as the set of elements that annotate the state S. Note that each state has an associated list which includes P elements (Pbeing, as stated before, the number of partial traces represented by state, S). On the contrary, in [88] the authors only annotate each state with a single attribute.

Considering definitions 6-13, we understand that the element, E_j, associated with a given partial trace, PT_j, is made up of the following components:

- Occurrence: $Occ(A_i,PT_j), \forall$ activities A_i in PT_j
- Cycle: $Cyc(A_i,PT_j), \forall$ activities A_i in PT_j
- Position: $Pos(A_i,PT_j), \forall$ activities A_i in PT_j
- Distance: $Dis(A_i,PT_j), \forall$ activities A_i in PT_j
- Duple: $Dup(A_i,A_k,PT_j), \forall$ activities A_i,A_k in PT_j
- Change(PT_j)
- Single(PT_j)
- Elapsed time: $Elt(PT_j)$
- Remaining time: $RT(PT_j)$

As an example, let us consider the following partial trace PT=<$A^{00}B^{06}$>, which is extracted from the first trace in Table 2.3. Subsequently, we understand that the element, E, associated with PT is made up of:

- Occurrence: $Occ(A,PT) = 1$, $Occ(B,PT) = 1$
- Cycle: $Cyc(A,PT) = 0$, $Cyc(B,PT) = 0$
- Position: $Pos(A,PT) = 1$, $Pos(B,PT) = 2$
- Distance: $Dis(A,PT) = 0$, $Dis(B) = 0$
- Duple: $Dup(A,A) = 0$, $Dup(A,B) = 1$, $Dup(B,A) = 0$, $Dup(B,B) = 0$
- Change=1
- Single=2
- Elapsed time: $Elt = 6$
- Remaining time: $RT(PT) = 12$

Therefore, element E will be the following vector:

$$[1,1,0,0,1,2,0,0,0,1,0,0,1,2,6,12]$$

Algorithm 1 Construction of an Extended Annotated Transition System (EATS)

```
    Input: TS = (S, PT, TR): Transition System (Def. 7);
    Output $EATS$: Extended Annotated Transition System
for each $s \in S$ do
    List(s) = ∅                                          ▷ Initialise Lists of Elements
end for
for each $pt \in PT$ do                                  ▷ For each partial trace
    $CS \leftarrow S(pt)$                                ▷ State associated to $pt$ (Def. 3)
    $E = \varnothing$                                    ▷ Initialise Element
    for $i$=1,...,$M$ do                                 ▷ Def. 14, M # attr. values of $pt$
        $E \leftarrow E \cup valueOfAtt_i(pt)$           ▷ Add $i-th$ attribute value
    end for
    $E \leftarrow E \cup RT_{pt}$                        ▷ Add remaining time of $pt$
    $List(CS) \leftarrow List(CS) \cup E$
end for
$List \leftarrow \{List(s), \forall s \in S\}$
$EATS \leftarrow (TS, List)$
return $EATS$
```

In Algorithm 1, we describe the procedure for formally building our Extended Annotated Transition System. Lines 1 to 3 initialise the Lists of Elements that are associated with each state of the EATS. The loop in (line 4) reiterates all partial traces, whilst in line 5 we obtain the state associated with each partial trace. Then, we store the attribute values in the element (lines 7-19) according to Def. 14. The element is completed by adding the remaining time of the partial trace (line 10) and stored in the list associated with the current state (line 11). The procedure is repeated throughout all the partial traces in the EATS and, finally, the Extended Annotated Transition System is returned.

Table 4.3: List of Partial Traces belongs to traces shown in Table 4.1, where T1 is $<A^3A^{18}B^{24}A^{30}>$, and T2 is $<A^7B^{10}B^{15}C^{22}C^{31}>$.

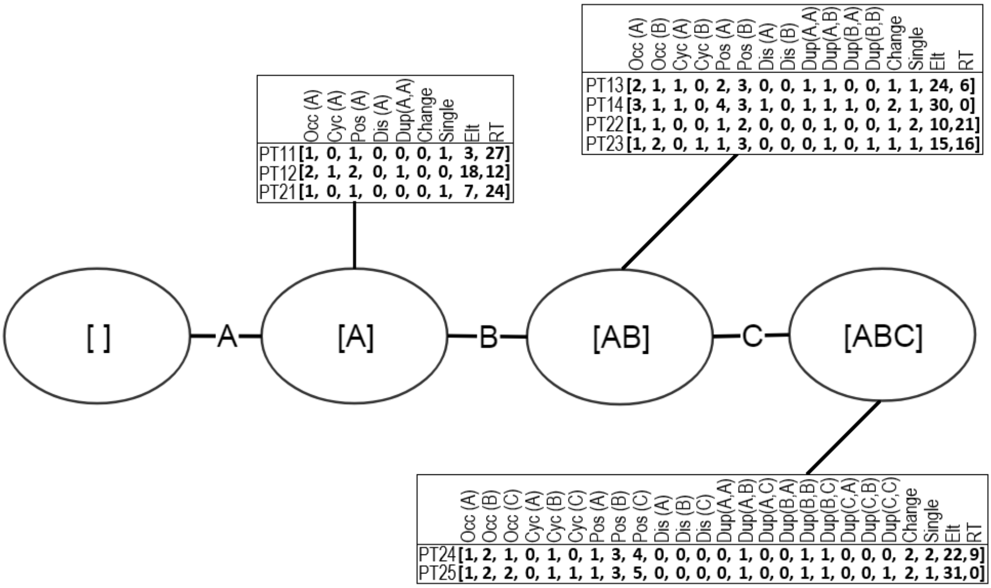
	T1	T2
Partial Trace	PT11: $<A^3>$ PT12: $<A^3A^{18}>$ PT13: $<A^3A^{18}B^{24}>$ PT14 (T1): $<A^3A^{18}B^{24}A^{30}>$	PT21: $<A^7>$ PT22: $<A^7B^{10}>$ PT23: $<A^7B^{10}B^{15}>$ PT24: $<A^7B^{10}B^{15}C^{22}>$ PT25 (T2): $<A^7B^{10}B^{15}C^{22}C^{31}>$

In Figure 4.2, we show an example of building and annotating the EATS which corresponds to two sample traces shown in Table 4.3. For instance, state $S = [A]$ is the state associated with the following three partial traces: PT11:=$<A^3>$, and PT12:= $<A^3A^{18}>$, (from $T1$), and PT21:= $<A^7>$, (from $T2$) as listed in Table 4.3. Therefore, we have an associated list of three elements made up of the corresponding eight attribute values defined in Section 4.3 and the corresponding remaining time for each partial trace (respectively 27, 12, 24).

Once the EATS is built and annotated with attributes values, linear regression is ready to be applied for each list of states in the EATS model, as we will discuss in the next section.

4.3.2 Obtaining the Linear Regression Model for the Estimations

Once the EATS is built and annotated, linear regression is applied to the list of each state, taking the values of the attributes as independent variables and the remaining time as the dependent variable. In this way, we obtain an expression for each state that will allow us to estimate the remaining time for any new trace represented by the same state. We will take note that each state, S, in the EATS is annotated with a list of elements (vectors) List(S), which includes the values of the attributes for all the partial traces represented by S. Each of these lists contains all the data (predictors or independent variables) needed for performing



	Occ (A)	Cyc (A)	Pos (A)	Dis (A)	Dup(A,A)	Change	Single	Elt	RT
PT11	1	0	1	0	0	0	1	3	27
PT12	2	1	2	0	1	0	0	18	12
PT21	1	0	1	0	0	0	1	7	24

	Occ (A)	Occ (B)	Cyc (A)	Cyc (B)	Pos (A)	Pos (B)	Dis (A)	Dis (B)	Dup(A,A)	Dup(A,B)	Dup(B,A)	Dup(B,B)	Change	Single	Elt	RT
PT13	2	1	1	0	2	3	0	0	1	1	0	0	1	1	24	6
PT14	3	1	1	0	4	3	1	0	1	1	1	0	2	1	30	0
PT22	1	1	0	0	1	2	0	0	0	1	0	0	1	2	10	21
PT23	1	2	0	1	1	3	0	0	0	1	0	1	1	1	15	16

	Occ (A)	Occ (B)	Occ (C)	Cyc (A)	Cyc (B)	Cyc (C)	Pos (A)	Pos (B)	Pos (C)	Dis (A)	Dis (B)	Dis (C)	Dup(A,A)	Dup(A,B)	Dup(A,C)	Dup(B,A)	Dup(B,B)	Dup(B,C)	Dup(C,A)	Dup(C,B)	Dup(C,C)	Change	Single	Elt	RT
PT24	1	2	1	0	1	0	1	3	4	0	0	0	0	1	0	0	1	1	0	0	0	2	2	22	9
PT25	1	2	2	0	1	1	1	3	5	0	0	0	0	1	0	0	1	1	0	0	1	2	1	31	0

Figure 4.2: Extended Annotated Transition System with lists of elements (attributes values) integrated. The traces considered here are the ones indicated in Table 4.1

the remaining time estimation. Therefore, we have a single dataset associated with each state. As the dataset is established, we are now in the position to describe our remaining time estimation model. Fundamentally, it is made up of linear regression functions that are obtained for each dataset (List). The independent variables of the regression functions are the values of attributes (elements) in the List, the remaining time being the dependent variable.

The method for obtaining the EATS and these linear regression functions that predict the remaining time for any new partial trace is described in Algorithm 2. The first step is to create the list of attributes with which each state of the transition system is annotated (lines 4-12): each partial trace, pt, is assigned to a transition system state, where all the attributes, $valueOfAtt_i(pt)$, are extracted from the structural analysis of pt, thus obtaining a vector of attributes E. Once all the partial traces associated with state, s, are processed, a list of vectors $List(s)$ is obtained. $List(s)$ is the input to the linear regression technique that is applied to each state (lines 14-17). Finally, the set of linear regression functions are returned.

Algorithm 2 Obtaining the Linear regression functions in the EATS

Input: $TS = (S, PT, TR)$: Transition System, where S is the set of states, PT is the set of partial traces, and TR is the transition relation

Output LR_{TS}: Linear Regression Functions for TS

```
for each $s \in S$ do
    List(s) = ∅                                        ▷ Initialise Lists of Elements
end for
for each $pt \in PT$ do                                ▷ For each partial trace
    $CS \leftarrow S(pt)$                              ▷ State associated to pt
    $E = \varnothing$                                  ▷ Initialise Element
    for $i$=1,...,$M$ do                               ▷ M is # attr. values of pt
        $E \leftarrow E \cup valueOfAtt_i(pt)$
    end for
    $E \leftarrow E \cup RT_{pt}$                      ▷ Add remaining time of pt
    $List(CS) \leftarrow List(CS) \cup E$
end for
$List \leftarrow \{List(s), \forall s \in S\}$
for each $s \in S$ do
    $List(s) \leftarrow removeOutliers(List(s))$       ▷ Outliers removed from each list
    $LR_{TS} \leftarrow LR_s(List(s))$                 ▷ Regression model obtained for each list
end for
return $LR_{TS}$
```

4.4 Experimental validation

We have validated our model using ten real-life event logs of BPI Challenges 2012w, 2013, 2015 and 2017, and Hospital Bill and Traffic Fine [1]. Logs in these datasets come from very different fields of applications, such as administrative and financial processes, the billing of medical services and road traffic fines management, which are considered a de-facto benchmark used for Research Challenges in the business process management area. Table 4.4 describes these event logs in more details.

4.4.1 Comparison with the baseline work

We compared our approach to the baseline ATS model described in [88]. To evaluate the results of our prediction model, we use the following usual three metrics to measure the error between the real remaining time and the predicted remaining time using our method:

Table 4.4: Description of the 10 real-life event logs used for validation.

Event logs	Description	# Cases	# Events
BPIC12w [92]	Application process for a personal loan or overdraft within a Dutch Financial Institute	8,723	60,780
BPIC13 [77]	Handling Incidents Process from Volvo IT of Belgium.	7,554	57,742
BPIC15 [93] (5 logs)	All building permit applications over four years provided by three Dutch municipalities.	1,199 832 1,409 1,053 1,156	52,217 44,354 59,681 47,293 59,083
BPIC17 [94]	Application process for a personal loan or overdraft within a Dutch Financial Institute	31,509	41,862
Hospital Billing [50]	Billing financial data from the ERP system of a Dutch hospital.	100,000	451,359
Traffic Fine [51]	Data from an information system managing road traffic fines.	150,370	561,470

- Accuracy of the prediction quality by calculating the difference between the real remaining time (RT) and the predicted remaining time (PT).

$$Accuracy = (1 - (|RT - PT|/RT)) * 100\% \tag{4.1}$$

- Mean Absolute Error

$$MAE = 1/n \sum_{i=1}^{n} |b_{\mathrm{i}} - \overline{b}| \tag{4.2}$$

- Mean Absolute Percentage Error

$$MAPE = 1/n \sum_{i=1}^{n} (|b_{\mathrm{i}} - \overline{b}|)/b_{\mathrm{i}} \tag{4.3}$$

In Table 4.5, we can see that on average our model performs better in all ten real-life event logs for all of the three metrics considered. In more detail, we can see that in 28 out of the 30 cases (93.3%) our model performs better, only being outperformed by [88] in two of the MAPE results. The average difference in MAE is 35.36 days, with values ranging from 4.17 to 116.58 days, in all cases favourable to our method. Therefore, our approach produces better results than [88] in a consistent way for a variety of application areas.

Table 4.5: Accuracy, MAE (in days), and MAPE results of the approach compiled in (88) compared to our approach.

		BPIC	BPIC	BPIC	BPIC	BPIC	BPIC	BPIC	BPIC	Hosp-	Traf-
		12	**13**	**15_1**	**15_2**	**15_3**	**15_4**	**15_5**	**17**	**ital**	**fic**
Accur.	our app.	**0.65**	**0.78**	**0.90**	**0.91**	**0.92**	**0.93**	**0.94**	**0.51**	**0.66**	**0.75**
	[88]	0.39	0.41	0.44	0.48	0.47	0.55	0.56	0.17	0.35	0.50
MAE	our app.	**1.52**	**1.73**	**3.58**	**8.69**	**1.19**	**3.87**	**4.66**	8.75	**19.01**	**31.78**
	[88]	5.69	6.45	42.94	89.68	19.76	20.06	46.83	**6.28**	52.33	148.36
MAPE	our app.	**0.01**	2.85	**1.54**	**2.47**	**0.22**	**2.34**	**0.43**	**0.25**	7.59	**0.00**
	[88]	0.24	**2.79**	5.66	9.91	2.67	2.91	6.25	0.40	**7.00**	0.01

4.4.2 Comparison of MAE metric using Earliness

Once we have shown that our approach improves the results in [88], we extend our experimentation very significantly by comparing our model to another sixteen state-of-the-art models described in [97], which includes non-ATS, ATS and Deep Learning models. We will use the same experimental setup and conditions described in [97] to obtain comparable results. Such conditions are: *(i)* no dataset pre-processing was made, *(ii)* traces are sorted depending on the start time, *(iii)* the prefix[1] length of the traces was 20 activities. (i.e., the first 20 activities were considered for the remaining time estimation), and *(iv)* outlier detection is applied to the training set.

The experimental results are described in Table 4.6, where we can see that our approach performs better in eight out of ten event logs. But we should consider that the MAE metric reported in [97] is a specific average MAE, where longer (less frequent) prefixes get lower weights (since not all the traces necessarily reach that length) and shorter prefixes (more frequent) get higher weights. Therefore, this weighting scheme is favourable for shorter prefixes which, in principle, produces a bias for these cases that benefits the results of, for instance, machine-learning approaches. For evaluating our model, we include all the prefixes and provide the MAE, not an average MAE. This means we are considering all traces and do not introduce a bias towards the shorter traces. In spite of these conditions, which are unfavourable to our method, we see that we outperform the results reported in [97] in 80% of the event logs considered.

Another key point to highlight about the results in Table 4.6 is that, although our method

[1] Prefix log is a custom log that includes n number of activities in each trace of the original log. If $n = 20$, then the length of each trace in the prefix log is 20 or less if the trace length is lower. [97]

performed better on average, the standard deviation is very high in general, and therefore the dispersion of data is also high, as well as the range of errors. Therefore, this suggests that improvements in the model should be made in order to reduce the high-standard deviation values. This is accomplished in Chapter 5.

We conclude that the EATS approach makes sense since it improves the ATS baseline very much. Therefore, this provides an empirically tested initial base for our model. Nevertheless, one of the limitations (or threats to validity) of our model is that the partial traces in the list can be very diverse, and in some real cases accuracy could be very low. This could happen in complex datasets where the values of the attributes may be very similar (or even identical) for some partial traces, but the corresponding remaining times could be (very) different among themselves. This situation is likely to occur to some extent in the realm of Business Process Management. For some real applications, it may be also the case that the accuracy/error we have achieved (although much better than [88]) could not be good enough and may need to be improved in order to be used in real practice. Furthermore, results are also better than those reported by all the models described in [97], although the high range of standard deviation of our prediction that appears in Table 4.6 should be taken into consideration in order to produce a new improved model. Because of this, although the results obtained with our basic model are promising, in the next chapter we will explore an enhanced version following the idea of partitioning the list of partial traces associated with each state, with the aim to calculate a regression expression for each partition. In this way, for each of the states in the EATS, we will have more than one regression expression which is likely to better adapt to complex datasets and further improve the promising accuracy results we have presented in this chapter.

Table 4.6: MAE results (in days) of the 16 approaches compiled in (97) compared to our approach based on the conditions of (97), in the first row). Symbol '-' in some cells means this value is not provided by the authors.

Technique	BPIC2012w	BPIC2013	BPIC2015_1	BPIC2015_2	BPIC2015_3
Earliness	**4.650 ± 4.260**	**4.150 ± 3.100**	**22.330 ± 46.330**	**40.530 ± 75.160**	**16.910 ± 40.550**
TS	7.505 ± 1.036	-	56.498 ± 8.341	118.293 ± 16.819	26.412 ± 8.082
LSTM	6.344 ± 0.994	-	39.457 ± 5.708	61.620 ± 2.061	19.682 ± 2.646
SPN	8.538 ± 0.772	-	66.509 ± 17.131	81.114 ± 8.033	26.757 ± 10.378
FA	6.946 ± 1.057	-	-	-	-
cluster_agg	7.180 ± 0.953	-	40.705 ± 1.824	68.185 ± 2.649	23.087 ± 3.226
cluster_index	7.074 ± 1.254	-	38.092 ± 2.988	66.957 ± 3.436	24.497 ± 1.887
cluster_last	7.061 ± 1.019	-	38.388 ± 3.478	62.781 ± 2.347	22.544 ± 1.656
prefix_agg	7.260 ± 0.935	-	46.765 ± 23.581	71.210 ± 8.893	24.152 ± 2.785
prefix_index	7.155 ± 0.942	-	37.525 ± 2.746	66.883 ± 3.756	21.861 ± 3.292
prefix_last	7.139 ± 0.851	-	37.975 ± 5.903	64.708 ± 5.749	23.574 ± 3.778
noBucket_agg	7.082 ± 1.020	-	35.962 ± 3.744	67.914 ± 2.467	24.453 ± 3.577
noBucket_index	6.982 ± 1.340	-	35.451 ± 2.499	65.505 ± 3.442	23.025 ± 1.587
noBucket_last	7.021 ± 1.099	-	37.442 ± 3.607	64.110 ± 2.332	25.150 ± 1.271
state_agg	7.465 ± 0.622	-	42.949 ± 2.725	68.768 ± 4.094	28.427 ± 9.844
state_index	7.510 ± 0.585	-	-	-	-
state_last	7.539 ± 0.554	-	42.946 ± 2.691	68.296 ± 3.762	27.826 ± 8.280

Technique	BPIC2015_4	BPIC2015_5	BPIC2017	Hospital Bill	Traffic Fine
Earliness	**22.030 ± 37.470**	**18.960 ± 39.570**	**4.720 ± 6.900**	42.210 ± 37.150	196.660 ± 175.560
TS	61.630 ± 5.413	67.699 ± 7.531	8.278 ± 2.468	46.491 ± 21.344	190.949 ± 15.447
LSTM	48.902 ± 1.527	52.405 ± 3.819	7.150 ± 2.635	**36.258 ± 23.870**	**178.738 ± 89.019**
SPN	-	51.202 ± 5.889	10.731 ± 0.370	71.377 ± 29.082	193.807 ± 96.796
FA	-	-	-	51.689 ± 14.945	223.808 ± 14.859
cluster_agg	51.555 ± 2.363	45.825 ± 3.028	7.479 ± 2.282	42.934 ± 26.136	210.322 ± 98.516
cluster_index	56.113 ± 6.411	44.587 ± 4.378	-	-	209.139 ± 98.417
cluster_last	51.451 ± 4.189	46.433 ± 4.085	7.457 ± 2.359	48.589 ± 26.708	208.599 ± 99.549
prefix_agg	53.568 ± 6.413	46.396 ± 2.466	7.525 ± 2.306	43.060 ± 25.884	201.614 ± 99.484
prefix_index	50.452 ± 4.605	44.290 ± 3.669	7.421 ± 2.360	41.698 ± 25.944	209.085 ± 99.708
prefix_last	53.053 ± 5.665	46.639 ± 3.718	7.482 ± 2.325	48.528 ± 26.714	209.304 ± 102.027
noBucket_agg	54.890 ± 1.894	49.203 ± 1.833	7.437 ± 2.381	43.483 ± 25.000	211.017 ± 93.198
noBucket_index	52.282 ± 1.182	50.153 ± 1.097	-	-	208.879 ± 92.250
noBucket_last	56.818 ± 1.729	49.027 ± 1.954	7.525 ± 2.244	50.496 ± 23.961	204.758 ± 93.399
state_agg	49.318 ± 2.699	49.873 ± 2.658	-	43.835 ± 25.984	211.439 ± 98.351
state_index	-	-	-	41.095 ± 26.499	210.408 ± 99.276
state_last	49.038 ± 2.498	49.556 ± 2.575	7.521 ± 2.341	48.902 ± 27.001	209.206 ± 100.632

CHAPTER 5

AN ENHANCED MODEL INCLUDING LISTS PARTITIONING

5.1 Introduction

In Chapter 4, we presented our ATS-based model, and the result we have in Section 4.4 show that our model outperforms the Van der Aalst baseline [88]. Nevertheless, as described in Chapter 3, there are other models in the literature which do not follow the ATS-based model as described in [8, 71, 98, 95, 78, 13, 31] and in fact obtain more accurate results than this baseline.

In the literature, we found that some non-ATS models outperform the baseline work [88], for instance the LTSM [78], SPN [71] and FA [98] models, as well as the model used in [97]. On the other hand, we found that the LTSM model [78] outperforms our work specifically in BPIC 17. Therefore, we enhanced our work by applying the partitioning technique, not only to perform better in BPIC 17 but also in any other real-life logs.

As discussed in the final remarks of Chapter 4, the main reason that the remaining time estimation models produce estimations which could be considered to be not very good in some realms of application is that the (partial) traces in the ten real log datasets we have considered vary greatly in terms of size, number of activities and execution times. This is usually the general scenario for real business process data, such as administrative procedures or applications, industrial incidents management or processes in a hospital or other big organisations/institutions [92, 94]. In many cases, the remaining time values range is vast (e.g.

from a few seconds to a hundred thousand seconds) even for traces that are very similar or even identical.

In order to face this problem, in this chapter, we will enhance the basic model presented in Chapter 4 by introducing a data partitioning technique of the partial traces list associated with each state in the EATS. Firstly, we will further prompt this issue through a simple experimental example. Secondly, we will define the partitioning procedure and how the remaining time is calculated in the enhanced model for a new trace. Finally, we will experimentally validate the enhanced model, showing how it performs better than the other state-of-the-art models (not only the ATS-based models).

5.2 Estimation of the Remaining Time: Dataset partitioning

Before describing the details of the regression model we use for estimating the remaining time, we should take into account the following considerations. Let us recall, firstly, that according to the EATS model, each state, S, in the EATS is annotated with a list of elements (vectors) $List(S)$ which include the values of the attributes for all the partial traces represented by S. Each of these lists contains all the data (predictors or independent variables) needed for performing the remaining time estimation. Therefore, we have a single dataset associated with each state.

Applying the linear regression technique to each of these datasets as described in Chapter 4 (basic model) is likely to produce poor estimations. Furthermore, as previously discussed, the basic model may not be sufficient for exceeding the results of the other state-of-the-art models. In order to better explain this, we will present in the following section a simple example that illustrates this limitation in our basic model.

Our simple illustrative example uses the real case described in the BPIC12w dataset, which contains logs taken from a Dutch Financial Institute [92]. The process of this real-life event log is an application process for a personal loan. We used only the process related to the work item belonging to the application.

In Table 5.1, we show the accuracy results after applying a linear regression method to the whole dataset associated with each state. We can see that the accuracy values range from 0.23 to 0.53, which are acceptable accuracy values when compared to the accuracy of the baseline work [88], but perhaps not good enough for certain application fields. Since huge variability is a usual feature in real cases, we have endowed our estimation method with

Table 5.1: Accuracy values for the set of partial traces associated with some states taken from a real business process described in (92) using our basic model described in the previous chapter.

	Accuracy
States	Non-partitioned
{A}	0.53
{AB}	0.50
{ABC}	0.23
{E}	0.37
{AE}	0.39
{ABCF}	0.34

a dataset partitioning stage, which is described below. In Section 5.3, we will revisit this illustrative example to assess the impact of the partitioning stage on the accuracy results.

Our partitioning method consists of building partitions that contain partial traces with similar remaining times. The similarity is expressed here in terms of a threshold value in the following way: Let us assume we have two partial traces, PT_1 and PT_2, with their corresponding remaining times, RT_1, and RT_2. Without losing generality, let us consider that $RT_1 \leq RT_2$. We will consider that PT_1 and PT_2 belong to the same partition if $RT_1/RT_2 > th$, where $th \in [0,1]$ is a predefined threshold. This condition states that both partial traces belong to the same partitions if the remaining time of the PT_1 is above a given percentage (th) of the PT_2. On the contrary, if $RT_1/RT_2 \leq th$, PT_1 and PT_2 will belong to different partitions, since in this case their quotient is below the predefined threshold. For example, for two partial traces PT_1 and PT_2 and a threshold value 0.40, this means that if the remaining time of PT_1 is above 40% of PT_2 they will be grouped in the same partition (their remaining times are considered similar). Otherwise (if below or equal 40%), they will be grouped in different partitions (their remaining times are considered different).

In Algorithm 3 we present a detailed description of our Threshold-Based Partitioning (TBP) procedure for a given state, s, and a threshold, th. In the first place (line 1) the List associated with s is ordered accordingly to the remaining time of all the partial traces associated with s. Partition building is described in the for loop (lines 4-10). In line 6, this condition is formalised as previously indicated. When it holds, a new partition is started (lines 7-8). When it does not, the for loop in line 4 continues to reiterate throughout all the partial traces and groups them in the same partition (line 5). Finally, the last partial trace is assigned (line 11)

Algorithm 3 Threshold-based partitioning (TBP) of the Partial Traces (List) associated with a state, s, in the EATS

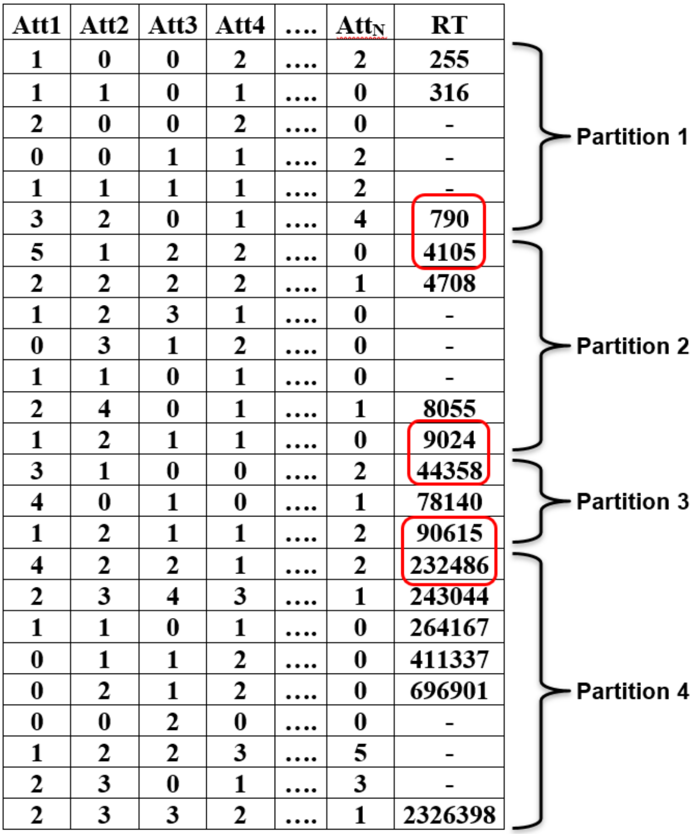
```
    Input: $s$, a state of the EATS (Def. 14); $th \in [0,1]$ a partition threshold
    Output $PL = \{P_1, .., P_{\mathrm{n}}\}$: Partitions List
 1: $SortedList(s) \leftarrow List(s)$ sorted in ascending order by the Remaining Time $RT_p$ of its
    elements $E_p$, $p = 1, ..., P$ (Def. 14)
 2: $n = 1$                                              ▷ Partition 1
 3: $P_n = \varnothing$                                  ▷ Initialise Partition 1
 4: for $p = 1, ..., P-1$ do                             ▷ $P$: List size
 5:     $P_n \leftarrow P_n \cup E_p$
 6:     if $RT_p / RT_{p+1} \leq th$ then                ▷ Abrupt change: new partition
 7:         $n \leftarrow n + 1$                         ▷ New partition
 8:         $P_n = \varnothing$                          ▷ Initialise the new partition
 9:     end if
10: end for
11: $P_n \leftarrow P_n \cup E_p$                        ▷ Last Element in the List
12: return $PL : \{P_1, ..., P_n\}$
```

to its corresponding partition: a new partition in case the threshold condition was met and the last partition in case it did not.

One key issue here is the number of intervals (or threshold points) to be defined (i.e. if we want to create very close intervals or wide intervals). It is important to define the number of segments in a balanced way, for example not to define it as very low, since the range values for the estimation time will be high and the accuracy will be lower, but also not to define it as very high, since this will increase the computational cost and cause over-fitting. In Chapter 6, we will discuss this issue again from a quantitative pragmatical point of view, in order to provide a range of threshold values with a right balance between low accuracy and over-fitting.

In Figure 5.1, we show an example of applying the Threshold-based Partitioning Procedure (Algorithm 3 for a threshold value $th = 0.4$.) We highlight the quotient values that fulfil the threshold condition (Algorithm 3, line 6) and therefore define the limits of each partition. All of these quotients are less than the 0.4 threshold, respectively:

- 790/4,105=0.19
- 9024/44,358=0.20
- 90,615/232,486=0.39



Att1	Att2	Att3	Att4	....	Att$_N$	RT
1	0	0	2	....	2	255
1	1	0	1	....	0	316
2	0	0	2	....	0	-
0	0	1	1	....	2	-
1	1	1	1	....	2	-
3	2	0	1	....	4	790
5	1	2	2	....	0	4105
2	2	2	2	....	1	4708
1	2	3	1	....	0	-
0	3	1	2	....	0	-
1	1	0	1	....	0	-
2	4	0	1	....	1	8055
1	2	1	1	....	0	9024
3	1	0	0	....	2	44358
4	0	1	0	....	1	78140
1	2	1	1	....	2	90615
4	2	2	1	....	2	232486
2	3	4	3	....	1	243044
1	1	0	1	....	0	264167
0	1	1	2	....	0	411337
0	2	1	2	....	0	696901
0	0	2	0	....	0	-
1	2	2	3	....	5	-
2	3	0	1	....	3	-
2	3	3	2	....	1	2326398

Partition 1
Partition 2
Partition 3
Partition 4

Figure 5.1: An example of a dataset sample showing Partitions and Threshold points.

Thus indicating that the corresponding remaining times are not similar and therefore will be grouped in different partitions. These partitions are the four indicated in Figure 5.1, labelled Partition1, ..., Partition 4. After partitioning, this dataset splits into four partitions and, therefore, it is ready for the linear regression model to be applied to each partition, as described in the following section. As pointed out before, four regression expressions will be obtained, one for each partition.

5.3 Linear Regression Model

Now that the need for partitioning the dataset has been established and our partitioning strategy has been described, we are in a position to describe our remaining time estimation model. It is fundamentally made up of linear regression functions that are obtained for each of the partitions of the previously described dataset. The independent variables of the regression functions are the values of attributes (elements) in each partition, the dependent variable being the remaining time.

A simple illustrative example showing the impact of this partitioning stage in the remaining-time estimation is shown in Table 5.1 (which was introduced in the previous section). Now, in the 'Partitioned' column, we present the accuracy results after applying the linear regression method to each of these partitions. When comparing these results with the ones in the 'Non-partitioned' column, we can see an important improvement in the accuracy results, which now range from 0.67 to 0.78. On the basis of this example, we present in the next section a detailed validation of our method.

The details of the remaining time estimation for any new trace, PT NEW, are described in Algorithm 4. Once the state associated with PT_{NEW} is obtained (line 1), its partition list $PL = \{P_1, ..., P_n\}$ is returned by Algorithm 3 (line 2). Also, the associated vector of attributes of the new trace is obtained (as indicated in Def. 14) and stored in the corresponding Element. The algorithm basically searches for the Partial Traces in PL, which are the closest ones to PT_{NEW} in terms of the Manhattan distance of the values of their attributes, as indicated in Def. 14. Searching for these Partial Traces and obtaining their associated partitions is done in lines 5-15. Once the Partitions are obtained, their indexes are stored in $partitionIndex$ and their corresponding linear regression functions $\{RL_k, k \in partitionIndex\}$ are applied to the values of their attributes of PT_{NEW}. The average of these results is returned as the estimation of the remaining time of our model.

After we constructed the Threshold-based partitioning (TBP), we applied it initially on the BPIC12w event log, the same one that we used in Table 5.1, and showed how the partitioning technique enhances the prediction accuracy in Table 5.2, where the average of accuracy for non-partitioned is 0.39, and for the partitioned is 0.72. The results show the difference between each state, which is clearly improved by the Threshold-based partitioning (TBP) technique.

Algorithm 4 Time Prediction Model (TPM) of a new trace

```
Input: $PT_{NEW}$: new Partial Trace
Output $PRT$: Predicted Remaining Time for $PT_{NEW}$.
$S \leftarrow S(PT_{NEW})$                                        ▷ State which represents $PT_{NEW}$
$PL = \{P_1,...,P_n\}$:= Partitions List associated with $S$, as returned by Algorithm 3
$RL = \{R_1,...,R_n\}$:= List of Linear Regression functions associated with $PL$, as indicated in Section 5.2
$E_{NEW}$ := Element associated with $PT_{NEW}$                    ▷ (Def. 14)
$distanceMin \leftarrow +\infty$
for $k = 1,...,n$ do                                               ▷ For all partitions in $PL$
    for each $PT \in P_k$ do
        $$dist = \sum_{m=1}^{M} |valueOfAtt_m(PT) - valueOfAtt_m(PT_{NEW})|$$
        if $dist < distanceMin$ then                               ▷ New min
            $partitionIndex = \{k\}$;
            $distanceMin \leftarrow dist$
        else if $dist == distanceMin$ then
            $partitionIndex = partitionIndex \cup \{k\}$;
        end if
    end for
end for
$PRT \leftarrow$ Average of the estimations obtained with the regression models $\{RL_k, k \in partitionIndex\}$ applied to $E_{NEW}$
return $PRT$
```

5.4 Experimental validation

In this chapter, now that we have justified and defined the partitioning technique, we will show our extensive experimental results in three different ways: *i)* Results considering different threshold values, *ii)* Threshold choice, and *iii)* Comparison to other approaches.

5.4.1 Results considering different threshold values

In this section, we present the validation results of our approach for different threshold values of the partitioning strategy (TBP) described in Section 5.2. Our method is compared to the

Table 5.2: Accuracy values for the set of partial traces associated with some states taken from a real business process described in (92) following two strategies: Non-partitioned (the basic model described in the previous chapter) and Partitioned (the enhanced model described in this chapter).

	Accuracy	
States	Non-partitioned	Partitioned
{A}	0.53	0.78
{AB}	0.50	0.77
{ABC}	0.23	0.67
{E}	0.37	0.70
{AE}	0.39	0.73
{ABCF}	0.34	0.69

ATS baseline approach described in [88] for the ten datasets in Table 4.4. This validation aims to provide a general overview of the dependence of our remaining time estimation results on the TBP threshold values. We used two metrics to compare our results to [88]: the Mean Absolute Error (MAE) to measure the error between real remaining and the predicted remaining time and the Accuracy to assess the regression quality in objective terms. According to [97], using RMSE as a metric should be avoided in this context, since it is very sensitive to outliers. For this analysis of the threshold dependence, datasets were pre-processed for removing those values that are more/less than twice the standard deviation from the average.

Table 5.3: Comparison between our approach and (88), using the ten event logs described in Table 4.4. Here we show the MAE measurement values at each threshold point.

	Mean Absolute Error (MAE)									
Logs	BPIC12w	BPIC13	BPIC15_1	BPIC15_2	BPIC15_3	BPIC15_4	BPIC15_5	BPIC17	Hospital Bill	Traffic Fine
Aalst et al.[88]	5.690	6.450	42.940	89.680	19.760	20.060	46.830	6.280	52.330	148.360
Threshold										
0	1.398	1.587	3.850	8.738	6.509	4.558	5.956	5.012	13.194	22.465
0.25	1.486	1.629	3.290	9.248	5.540	4.667	5.502	3.370	12.847	20.958
0.5	1.313	1.497	3.070	8.715	6.148	4.228	5.956	3.661	11.806	20.574
0.7	1.695	1.611	3.560	8.495	5.520	4.194	5.787	3.600	14.931	22.911
0.75	1.256	1.520	3.320	9.676	6.121	4.258	5.394	3.879	13.194	19.335
0.9	1.189	1.442	3.000	8.044	5.910	4.166	5.208	4.687	13.310	24.290
0.925	**1.146**	1.381	3.150	7.234	4.954	3.492	4.745	5.671	13.194	20.958
0.95	1.154	**1.134**	3.080	7.778	**4.281**	3.314	4.271	5.557	**11.227**	23.842
0.975	1.256	1.846	**2.070**	**6.296**	4.471	**3.263**	**3.623**	**2.863**	12.153	**18.326**
Average ± STDEV	1.322 ± 0.18	1.516 ± 0.20	3.154 ± 0.49	8.247 ± 1.04	5.495 ± 0.78	4.016 ± 0.53	5.160 ± 0.80	4.255 ± 1.01	12.873 ± 1.07	21.518 ± 2.01

In Table 5.3, we show the results of the comparison between our approach with attributes selection and the baseline approach [88] in terms of the ten considered real-life event logs and the Mean Absolute Error (MAE) metric. We provide the results of our approach for a number of thresholds in order to assess the general quality of the results as well as the dependency of specific thresholds values. In order to have a general view of the method, we also provide (Table 5.3, bottom row), for each dataset, the average results of our method for all the thresholds considered. According to this, the average value (even including the standard deviation) of our method outperforms [88] in all ten datasets.

Looking at all the thresholds values, we can see that in all of the 90 cases considered, our approach outperforms [88]. The average MAE of our method is 6.76 days, the average MAE of [88] being 43.84 days. The differences in MAE between our approach and [88] range from 2.02 days (in BPIC17) to 126.84 days (in Traffic Fine), 37.08 days being the average difference, all cases being favourable to our method.

Table 5.4: Comparison between our approach and (88), using the ten event logs described in Table 4.4. We show the accuracy measurement values at each threshold point.

	Accuracy									
Logs	BPIC12w	BPIC13	BPIC15_1	BPIC15_2	BPIC15_3	BPIC15_4	BPIC15_5	BPIC17	Hospital Bill.	Traffic Fine
Aalst et al.[88]	0.390	0.410	0.440	0.480	0.470	0.550	0.560	0.170	0.350	0.500
Threshold										
0	0.690	0.785	0.894	0.898	0.893	0.921	0.925	**0.563**	0.649	0.782
0.25	0.641	0.774	0.896	0.899	0.906	0.921	0.928	0.535	0.682	0.785
0.5	0.690	0.770	0.905	0.897	0.900	0.918	0.925	0.558	0.700	0.797
0.7	0.705	0.790	0.898	0.904	0.899	0.918	0.930	0.549	0.693	0.775
0.75	0.718	0.771	0.899	0.899	0.901	0.924	0.935	0.562	0.710	0.753
0.9	0.691	0.800	0.908	0.902	0.908	0.927	0.936	0.546	**0.730**	0.784
0.925	**0.756**	0.791	0.912	0.912	0.907	0.933	0.938	0.491	0.705	0.785
0.95	0.738	0.835	0.909	0.915	0.916	0.936	0.941	0.526	0.723	0.772
0.975	0.694	**0.843**	**0.920**	**0.929**	**0.923**	**0.940**	**0.950**	0.533	0.720	**0.806**
Average ± STDEV	0.703 ± 0.03	0.795 ± 0.03	0.905 ± 0.01	0.906 ± 0.01	0.906 ± 0.01	0.927 ± 0.01	0.934 ± 0.01	0.540 ± 0.02	0.701 ± 0.03	0.782 ± 0.02

In Table 5.4, we show the same comparison for accuracy. Again, the average value (also considering the standard deviation) of our method outperforms [88] in all ten datasets and in all of the 90 cases considered. For the datasets considered, the average accuracy of our method ranges between [0.54, 0.93], whilst [88] ranges between [0.17, 0.56]. The average accuracy of our method is 0.81, the average accuracy of [88] being 0.43.

5.4.2 Threshold choice

In this section, we discuss how to choose an appropriate threshold value which could be used, in general, for any new dataset. We will support the discussion with the experimental analysis and the results that we will later describe. In an initial analysis, it seems straightforward that the best threshold choice should be the most precise one, i.e., the one that produces the highest accuracy or lowest MAE. In order to experimentally determine which is, in general, the most precise threshold, we need to consider the results in Tables 5.3 and 5.4, rank them, and calculate the average rank through all the datasets and thresholds, for both MAE and Accuracy. These results are summarised in Table 5.5. According to this, the most precise threshold (MPT) is the one with the lowest ranks, which is 0.975 for both Accuracy and MAE.

In a second analysis, from looking at the results in Tables 5.3 and 5.4 we can observe that, in general, high threshold values produce better results (i.e. lower MAE and higher accuracy). This means that the granularity of the partitioning intervals is higher, and consequently, the number of operations involved also increases.

Table 5.5: The average rank and the standard deviation throughout all of the real event logs

	Accuracy		MAE	
Threshold	Avg rank	STDEV	Avg rank	STDEV
0	6.9	2.4	7.1	1.4
0.25	6.5	1.7	6.0	2.3
0.5	6.3	2.4	5.0	2.1
0.7	5.9	1.5	6.3	2.1
0.75	5.4	2.1	5.6	2.0
0.9	4.0	1.4	4.9	2.0
0.925	3.7	2.2	3.7	2.3
0.95	3.3	2.5	3.1	2.5
0.975	**2.2**	**2.1**	**2.4**	**2.5**

Taking this consideration into account, it also becomes evident that, generally, choosing the most appropriate threshold is not necessarily only a matter of choosing the most precise one, but the one with a good balance between precision in the results (generally associated with high thresholds) and the number of operations involved (generally associated with low thresholds). In order to determine a threshold value with a good compromise between these two opposite constraints, we will apply the One-Standard-Error Rule [43], which is a well-known model selection technique, commonly used in cross-validation, in which we choose the strictest model whose error is no more than one standard error above the error of the best model [38]. This technique can be described as follows: in case of different error measurements, we calculate the rank for each measurement, then we give the lowest error a low rank. After that, we calculate the average of the rank. Now for the lowest average rank, we add one standard deviation, therefore, we will have a new point (lowest avg. rank plus one standard deviation). We will then have a new line from the new point towards the average rank axis; if there any intersection with the other axis, then this will be the best value of that axis.

By applying this rule, we will provide experimental support for selecting a threshold value with a balanced compromise between precision and the number of operations involved and that can, therefore, be labelled also as an appropriate choice for any new dataset.

The One-Standard-Error Rule is applied to the ranking described in Table 5.5 and looks for the lowest threshold value whose average error is no more than one standard deviation above the error of the best model. The threshold value obtained following this procedure (BCT, Best Compromise Threshold) will exhibit a good compromise between precision and the number of operations involved.

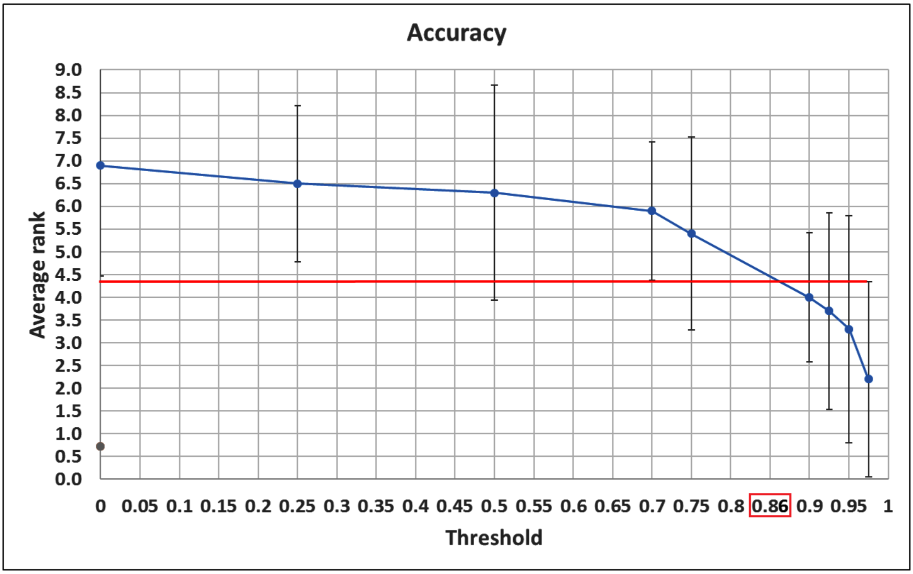


Figure 5.2: Model selection using the one-standard-error rule method (43) for the accuracy results in Table 5.5. The selected model corresponds to a 0.86 threshold.

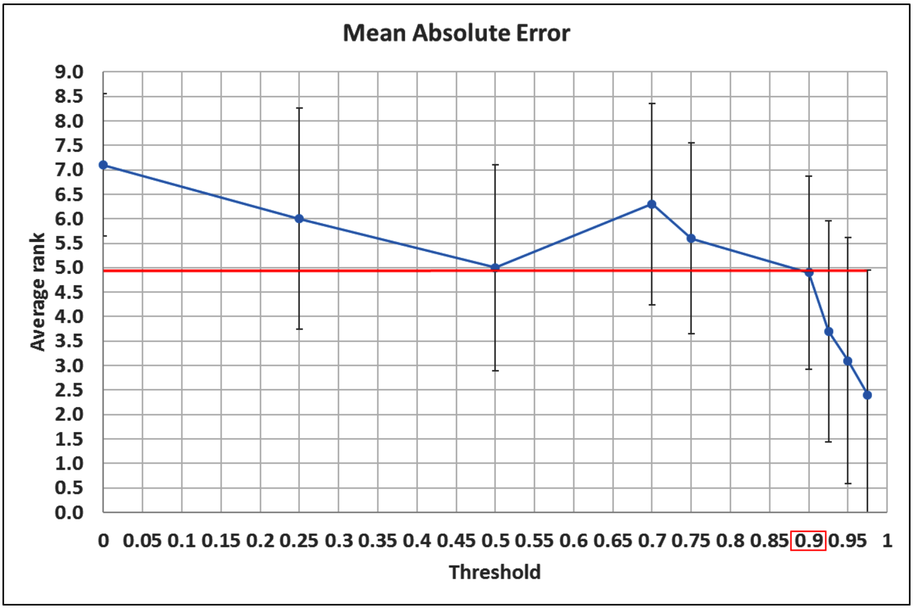


Figure 5.3: Model selection using the one-standard-error rule method (43) for the MAE results in Table 5.5. The selected model corresponds to a 0.90 threshold.

Figures 5.2 and 5.3 show the application of the One-Standard-Error Rule for both the accuracy and MAE results in Table 5.5. For both figures, the blue line indicates the average ranking values for each threshold considered. The red line indicates the rank value that corresponds to the best model in terms of average rank plus one standard deviation. The intersection of both lines indicates the Best Compromise Thresholds, 0.86 for Accuracy (Figure 5.2) and 0.90 for MAE (Figure 5.3) respectively, which are the recommended thresholds in terms of precision and the number of operations involved.

5.4.3 Comparison to other approaches using BCT and MPT

In this section we will provide two different experiments, i) the comparison of MAE metric between the two methods of threshold choice in Section 5.4.2 and 16 other state-of-the-art methods described in [97], and ii) the comparison of MAE metric between our method and 16 other state-of-the-art methods described in [97]. In this comparison, we will follow the same experiment setup.

Table 5.6: MAE results (in days) of the 16 approaches compiled in (97) compared to our approach in two different scenarios (BCT and MPT, in the first two rows). Symbol '-' in some cells means that this value is not provided by the authors.

Technique	**BPIC2012w**	**BPIC2013**	**BPIC2015_1**	**BPIC2015_2**	**BPIC2015_3**
BCT	2.390±2.810	**7.140±6.920**	4.000±18.990	6.670±23.470	1.280±7.390
MPT	**1.900±2.390**	7.280+7.780	**3.740±13.130**	**6.020±26.270**	**1.270±8.530**
TS	7.505 ± 1.036	-	56.498 ± 8.341	118.293 ± 16.819	26.412 ± 8.082
LSTM	6.344 ± 0.994	-	39.457 ± 5.708	61.620 ± 2.061	19.682 ± 2.646
SPN	8.538 ± 0.772	-	66.509 ± 17.131	81.114 ± 8.033	26.757 ± 10.378
FA	6.946 ± 1.057	-	-	-	-
cluster_agg	7.180 ± 0.953	-	40.705 ± 1.824	68.185 ± 2.649	23.087 ± 3.226
cluster_index	7.074 ± 1.254	-	38.092 ± 2.988	66.957 ± 3.436	24.497 ± 1.887
cluster_last	7.061 ± 1.019	-	38.388 ± 3.478	62.781 ± 2.347	22.544 ± 1.656
prefix_agg	7.260 ± 0.935	-	46.765 ± 23.581	71.210 ± 8.893	24.152 ± 2.785
prefix_index	7.155 ± 0.942	-	37.525 ± 2.746	66.883 ± 3.756	21.861 ± 3.292
prefix_last	7.139 ± 0.851	-	37.975 ± 5.903	64.708 ± 5.749	23.574 ± 3.778
noBucket_agg	7.082 ± 1.020	-	35.962 ± 3.744	67.914 ± 2.467	24.453 ± 3.577
noBucket_index	6.982 ± 1.340	-	35.451 ± 2.499	65.505 ± 3.442	23.025 ± 1.587
noBucket_last	7.021 ± 1.099	-	37.442 ± 3.607	64.110 ± 2.332	25.150 ± 1.271
state_agg	7.465 ± 0.622	-	42.949 ± 2.725	68.768 ± 4.094	28.427 ± 9.844
state_index	7.510 ± 0.585	-	-	-	-
state_last	7.539 ± 0.554	-	42.946 ± 2.691	68.296 ± 3.762	27.826 ± 8.280

Technique	**BPIC2015_4**	**BPIC2015_5**	**BPIC2017**	**Hospital Bill**	**Traffic Fine**
BCT	4.370±21.160	2.870±15.100	3.420±3.070	13.670±21.820	43.990±62.410
MPT	**2.790±14.370**	**2.230±12.640**	**3.270±2.840**	**12.850±20.690**	**33.310±58.350**
TS	61.630 ± 5.413	67.699 ± 7.531	8.278 ± 2.468	46.491 ± 21.344	190.949 ± 15.447
LSTM	48.902 ± 1.527	52.405 ± 3.819	7.150 ± 2.635	36.258 ± 23.870	178.738 ± 89.019
SPN	-	51.202 ± 5.889	10.731 ± 0.370	71.377 ± 29.082	193.807 ± 96.796
FA	-	-	-	51.689 ± 14.945	223.808 ± 14.859
cluster_agg	51.555 ± 2.363	45.825 ± 3.028	7.479 ± 2.282	42.934 ± 26.136	210.322 ± 98.516
cluster_index	56.113 ± 6.411	44.587 ± 4.378	-	-	209.139 ± 98.417
cluster_last	51.451 ± 4.189	46.433 ± 4.085	7.457 ± 2.359	48.589 ± 26.708	208.599 ± 99.549
prefix_agg	53.568 ± 6.413	46.396 ± 2.466	7.525 ± 2.306	43.060 ± 25.884	201.614 ± 99.484
prefix_index	50.452 ± 4.605	44.290 ± 3.669	7.421 ± 2.360	41.698 ± 25.944	209.085 ± 99.708
prefix_last	53.053 ± 5.665	46.639 ± 3.718	7.482 ± 2.325	48.528 ± 26.714	209.304 ± 102.027
noBucket_agg	54.890 ± 1.894	49.203 ± 1.833	7.437 ± 2.381	43.483 ± 25.000	211.017 ± 93.198
noBucket_index	52.282 ± 1.182	50.153 ± 1.097	-	-	208.879 ± 92.250
noBucket_last	56.818 ± 1.729	49.027 ± 1.954	7.525 ± 2.244	50.496 ± 23.961	204.758 ± 93.399
state_agg	49.318 ± 2.699	49.873 ± 2.658	-	43.835 ± 25.984	211.439 ± 98.351
state_index	-	-	-	41.095 ± 26.499	210.408 ± 99.276
state_last	49.038 ± 2.498	49.556 ± 2.575	7.521 ± 2.341	48.902 ± 27.001	209.206 ± 100.632

In Table 5.6, we show the comparison results for the MAE metric between our proposal and 16 other state-of-the-art methods described in a very recent survey [97], for the same ten datasets reported in the previous sections.

For a fair and more detailed comparison, we will provide the results of our method in two scenarios for the threshold selection: MPT provides the results obtained for the Most Precise Threshold (0.975) and BCT for the MAE Best Compromise Threshold (0.90) as defined in Section 5.4.2. It can be seen that our approach, even in the worst cases, produces the lowest error in all of the datasets. Compared with the best model reported in [97] (LSTM [78]), which is a deep learning-based approach, the average MAE of our MPT method is 7.49 days, the average MAE of LSTM being 50.02 days. Differences in MAE between our approach and the LSTM range from 3.88 days (BPIC2017) to 145.07 days (Traffic Fine), 38.28 days being the average difference that is favourable to our method.

We have also considered the impact of standard deviation, which is higher in our method than the others for most of the datasets considered. Comparing the worst case (MAE + STDEV), the average of our method is 35.59 days, the LSTM average being 58.25 days. Differences between our approach and the LSTM range within this metric from 3.05 days (BPIC 2012w) to 175.74 days (Traffic Fine), 35.59 days being the average difference, in all cases that are favourable to our method. According to these results, our method still outperforms LSTM when considering variance.

Apart from these experimental results, interpret-ability is also a key advantage of our method when compared to Deep Learning approaches, which are usually labelled as black-box approaches. Since our EATS approach is based on linear regression on attribute values that are related to the structure and contents of the trace, users can interpret and understand the variables' meaning and their relative importance (coefficients in the regression expressions). It should be taken into account that the interpret-ability of systems is an increasing demand in the context of Fairness, Accountability, Transparency and Ethics (FATE) in Artificial Intelligence and systems and applications in general.

From the previous experiments, we can see how our partitioning technique improves our result, not only performing in the baseline work [88] and other methods in the literature as we showed in Table 5.6, but also in terms of the results we have from our first model in Chapter 4.

However, the results that we have shown prove that our model performs other work, but the work that has been carried out in this chapter assumes that the number of events is always low. On the other hand, if the number of events is high, then a high number of attributes will be produced. In order to cope with this scalability problem, in Chapter 6 we will describe the use of an attribute selection method, which reduces the number of attributes by keeping in the model only the attributes that have a real impact on the remaining time estimation and removing the less relevant ones.

Chapter 6

Addressing scalability of the model

In the previous chapters, we have performed two types of experiments with the following aims:

- In Chapter 4, we compared our method with the baseline ATS-based proposal [88] in order to validate it in terms of accuracy and mean absolute error;
- In Chapter 5, we introduced the partitioning technique to enhance the results we have in the basic model in Chapter 4, proving how the prediction accuracy was improving for different threshold values and also obtaining experimental evidence about the influence of the threshold values in the results.

We also consider here the key issue that was previously highlighted in Section 4.3: that in general, a business process could involve a high number of activities, which would also mean that the number of attributes could also be high. Consequently, the number of operations related to the partitioning stage would also increase. Therefore, in order to keep our approach general, it is advisable to include an attribute selection method that reduces the number of attributes instances that were initially considered. Through our dealings with business process logs, we could have a process with a high number of activities and a huge number of attributes as we mentioned in Section 4.3.1.

The key issue we have faced in the model with the partition technique method is that where the many numbers of events increased, the number of attributes also increased. Regarding the nature of the real-life event logs, the traces in these event logs vary greatly in terms of size, number of activities and execution times. This is generally the usual scenario for real business process data, such as administrative procedures or applications, industrial incidents management or processes in a hospital or other big organisations/institutions [92, 94]. In many cases, the range of remaining time values is vast (e.g. from a few seconds to a hundred thousand seconds), even for traces that are very similar or even identical.

In addition to the huge number of attributes in the model, we also have another issue related to the huge number of partitions produced, especially for the high threshold value: if we have a high number of events and threshold values, then the model takes a long time to calculate the regression expression for each partition.

To explain more about the huge attributes number, let us recall the scenario from when the list was built.

Table 6.1: Sample log with three traces

Trace 1	$<A^3B^{10}B^{22}C^{40}A^{45}>$
Trace 2	$<A^2A^{16}C^{19}B^{24}B^{50}>$
Trace 3	$<A^7C^{11}C^{15}C^{24}B^{60}>$

For example, in Table 6.1 for state $\{$AB$\}$, we have two partial traces: PT12, $<$AB$>$, and PT13, $<$ABB$>$. Each vector in the list includes the corresponding 15 attributes values (plus the value to be predicted, i.e. the remaining time). With regard to the number of attributes, it is worth remembering that the number differs between the states, since it depends on how many activities engage in that state, as we have explained before. For instance, in this example, the partial traces associated with one event (State $\{$A$\}$) have eight attribute values, one for each of the definitions of Section 4.3. The partial traces associated with states $\{$AB$\}$ and $\{$AC$\}$ (two events each) have 15 attribute values each, since the existence of two activities doubles the number of Occurrence, Cycle, Position and Distance (one for each activity, 8 in total) and increases to $2^2 = 4$ the number of Duple values, plus other 3 attributes (Change, Single and Elapsed Time).

Assuming we have an event log with more activities as described in Table 6.2, it is clear that as the number of activities increases, the number of attributes also increases just as much and, in the case of a large number of events, the increase can be dramatic.

In order to keep our approach general and avoid scalability issues, in this chapter we will discuss the application of attribute selection methods which reduce the number of attributes considered in the model, thus only keeping the relevant ones. In this way, the number of operations involved and the computational cost of the model are reduced.

Table 6.2: Number of activities

# Activity	# Attributes
1	8
2	15
10	143
50	2703
100	10403

6.1 Scalability and the Attribute Selection Methods

Through our dealings with business process logs, we may come across a process with a high number of activities, with a huge number of attributes as we mentioned in Section 4.3.1.

The attribute selection problem is described as: given a set of candidate attributes, select a subset that performs the best under a certain classification system. This procedure can reduce not only the calculation time by reducing the number of attributes that the model needed, but in some cases it can also provide better classification accuracy due to finite sample size effects [104]. The term ‘attribute selection’ is taken to refer to algorithms that output a subset of the input attribute set. More general methods that create new attributes based on transformations or combinations of the original attribute set are termed attribute extraction algorithms.

Datasets with hundreds and thousands of attributes may cause a ‘dimensionality problem’. Moreover, some of the traditional classification and clustering algorithms cannot work correctly. One of the most practical techniques to cope with this problem is attribute reduction. Attribute reduction refers to the research of methods that have reduced dimensions present in the original data [46]. From a general point of view, there are two categories of attribute reduction, namely attribute selection (or variable selection) and attribute extraction (or attribute transform).

In this thesis, to cope with the dimensionality problem appearing in the previous model and to keep our approach general and avoid scalability issues, we recommend applying attribute selection methods to reduce the number of attributes and keep the relevant attributes. To achieve this, we will use several attribute selection methods

In some computational manner, certain attribute selection methods, for instance the best subset selection method, is not the preferred method to be applied if the data has a vast number of predictors. Such an attribute selection method could suffer from statistical complications in the case of a large number of predictors. If the search space is getting larger, then there will

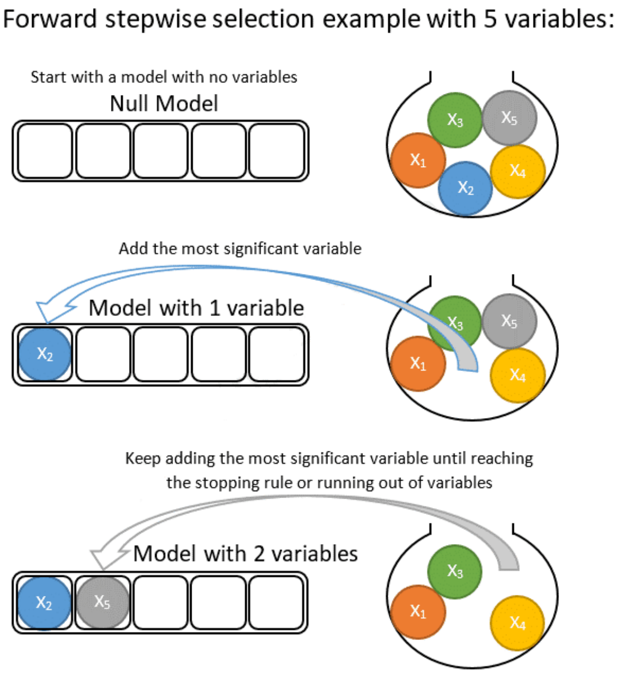

Forward stepwise selection example with 5 variables:
Start with a model with no variables
Null Model
X3
X5
X1
X2
X4
Add the most significant variable
Model with 1 variable
X2
X3
X5
X1
X4
Keep adding the most significant variable until reaching the stopping rule or running out of variables
Model with 2 variables
X2
X5
X3
X1
X4

Figure 6.1: Forward Greedy-Stepwise (16).

be a higher chance of getting models that are good with the training part. Even they aren't, they could still have a kind of predictive power on future data. Hence, a huge search space can lead to over-fitting and a high variance of the coefficient estimates [103].

For the previous two reasons, stepwise methods, which search a greatly more limited set of models, are more attractive choices to best achieve subset selection and other selection methods.

6.1.1 Forward Greedy-Stepwise

The first method we will use is *Forward Greedy-Stepwise* [103], which performs a greedy forward or backward search using the space of attribute subsets. It could start with an attribute or without attributes or from an arbitrary point in space. It stops when the addition or deletion of any remaining attributes results affects the decrease in evaluation. Figure 6.1 explains how it works.

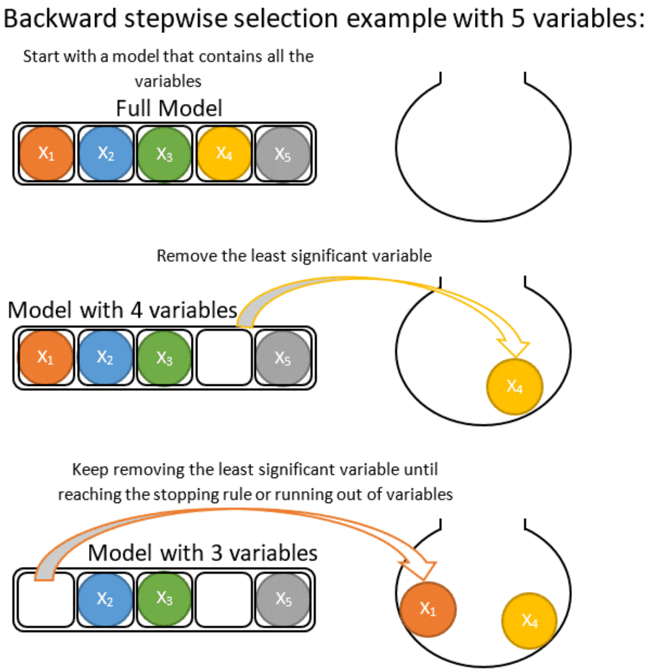

Backward stepwise selection example with 5 variables:
Start with a model that contains all the variables
Full Model
X1
X2
X3
X4
X5
Remove the least significant variable
Model with 4 variables
Keep removing the least significant variable until reaching the stopping rule or running out of variables
Model with 3 variables

Figure 6.2: Backward Greedy-Stepwise (16).

6.1.2 Backward Greedy-Stepwise

Along the same lines as the forward stepwise selection, the backward stepwise selection provides an efficient alternative to best achieve subset selection. Nevertheless, unlike the forward stepwise selection, it begins with the least full squares model containing all the predictors, and then iteratively removes the least useful predictors, one at a time [103]. Figure 6.2 explains how it works.

6.1.3 Forward Best-First

The second method is *Forward Best-First* [103], which searches the scope of attribute sets by greedily *hill-climbing*, expanded with a backtracking tactic. Best-First may begin with the empty set of attributes and search forward, start with the full set of attributes and search backward, or start at a random point and search in both directions, which considers all conceivable single attribute additions and deletions at a given point.

In the next section, we will demonstrate the result of our experiment. In experiments, we will use two attribute selection methods: i) Forward Greedy-Stepwise and ii) Forward Best-First. The reasons we exclude the Backward Greedy-Stepwise and use the Forward one refers to:

- The Forward Greedy-Stepwise starts with one attribute and is less costly in terms of computational load.
- We have done the experiments with the backward approach and found no differences in terms of the attributes selected.

6.2 Experimental validation

We have validated our model using eight real-life event logs of BPI Challenges 2012w, 2013, 2015, 2017, 2018, 2019, Hospital Bill, Traffic Fine and Credit Requirement [1]. Logs in these datasets come from very different fields of applications, such as administrative and financial processes, the billing of medical services and road traffic fines management, which are considered a de-facto benchmark used for Research Challenges in the field of business process management.

6.2.1 Experimental Setup

Each list of vectors that annotate each state is randomly divided into two parts: 80% for training and 20% for testing. We have considered all of the kinds of traces, independent of their size or structural features, removing from the analysis the lists which include a single vector. Furthermore, three measures have been used to compare the eight real-life logs. *Accuracy*, which calculates the difference between the real remaining time and the predicted one; *Mean Absolute Error* (MAE), defined as the arithmetic means of the prediction errors, and; *Mean Percentage Error* (MAPE), which measures error as the average of the unsigned percentage error.

6.2.2 Comparison with the baseline ATS model

We compared our approach with the baseline ATS model described in [88] to evaluate and discover if our model performs better in the two attribute selection methods. For reference, we also include the results with no attribute selection methods, which allows all of the attributes to be included in the linear regression.

In Table 6.4, we can see that, on average, our model performs better in all eight real-life event logs for all of the three metrics considered. In more detail, we can see that in 153 out of the 156 cases (98%) our model performs better, being outperformed by [88] only in three of the MAPE results. In MAE, with the values ranging from 3.21, low difference, to a high difference of 116.55 days, all cases are favourable to our method.

For the particular case of BPI15 event logs, we observe that using attribute selection techniques provides better accuracy than using the model without attribute selection. This dataset includes traces containing information on the building permit applications as well as objection procedures in various stages in five Dutch municipalities. This result suggests that some of the eight attributes may add some confusion to the regression model and that it is preferable that they are dropped. Therefore, the application of our model, in general, should consist of performing the attribute selection stage, not only for scalability and computational cost reasons, but also to aim to obtain better precision through discarding some attributes that may not be appropriate for a particular problem and, therefore, keeping only the most relevant and pertinent attributes.

6.2.3 Attribute Selection Methods

In this section, we provide a comparison between the attribute selection methods with the baseline ATS model. In Table 6.4, we show the results the eight real-life logs, including three error metrics with the two attribute selection methods we have explained in Section 6.1. Regarding accuracy, we can see that in general, we have the best accuracy values in the case of the *Forward Greedy-Stepwise*, and in the case of *Forward Best-First* we have the lowest Accuracy. The lowest MAE and MAPE values we are getting when no attribute selection is applied and increased when we apply the *Forward Greedy-Stepwise* and decreased when we applied the *Forward Best-First*. It is normal to have the same behaviour for all error metrics, which is expected as the higher number of attributes, the better the result. We have shown the difference in the attributes number for each method in Table 6.3, and we have shown the number of attributes that are included in the Linear regression.

In Table 6.5, it clearly appears that without any attribute selection methods, the result of the accuracy and MAE perform better, and with regard to MAPE, it almost equals the best results of the *Forward Greedy-Stepwise* method.

Table 6.3: Number of attributes for the different selection methods

	Event log:	**BPIC12w**	**BPIC13**	**BPIC15_1**	**BPIC15_2**	**BPIC15_3**	**BPIC15_4**	**BPIC15_5**
	# Activities:	6	3	26	26	26	26	26
	Attribute Selection Method							
# Attributes	None	527	93	461712	360868	1797584	903866	911422
	Forward Greedy-Stepwise	92	29	8227	6252	22527	12446	12373
	Forward Best-First	**55**	**13**	**6107**	**5040**	**19145**	**10375**	**10021**

	Event log:	**BPIC17**	**Hospital Bill**	**Traffic Fine**	**Credit Requirement**	**BPIC18**	**BPIC19**	
	# Activities:	6	10	8	7	32	31	
	Attribute Selection Method							
# Attributes	None	839	19605	907	193	489085	124278	
	Forward Greedy-Stepwise	133	1431	98	11	20747	8945	
	Forward Best-First	**95**	**775**	**62**	**10**	**11791**	**6118**	

From the previous analysis of Table 6.4 and 6.5, we can conclude that with more representation of related information attributes, we always have a better result than using the attribute selection methods, where these two methods decrease the number of attributes, and thus provide less information about each partial trace. Therefore, our experiments prove our hypotheses that collecting as much related useful information about the process as possible will improve the quality of the prediction of a running time of a business process.

On the other hand, we should consider the calculation time of the estimation. We show this in Table 6.5, where we include the time consumed for the attribute selection, the time for obtaining the regression expressions and for calculating the estimation time. The lowest calculation time on average is the *Forward Best-First* method. From the balanced point of view, Forward Best-First provides the lowest calculation time of the remaining time estimation, which is better than the *Forward Greedy-Stepwise*, and with no attribute selection method. And on the other hand, it provides an acceptable prediction time accuracy compared with the baseline approach of [88]. *Forward Best-First* method not only provides the fastest method to calculate the remaining time estimation but also less usage of the resources during the estimation process.

Table 6.4: Comparison between our approach and the model in (88) using different selection methods

	Event log:	BPIC12w	BPIC13	BPIC15_1	BPIC15_2	BPIC15_3	BPIC15_4	BPIC15_5
	Attribute Selection Method							
Accuracy	None	**0.68**	**0.65**	0.76	0.81	0.77	0.86	0.87
	Forward Greedy-Stepwise	0.66	0.64	0.80	0.84	0.80	0.88	0.89
	Forward Best-First	0.62	0.64	**0.84**	**0.86**	**0.83**	**0.90**	**0.91**
	[88]	0.39	0.41	0.68	0.68	0.74	0.71	0.72
MAE	None	**1.42**	**3.00**	5.13	11.33	2.69	8.35	7.54
	Forward Greedy-Stepwise	1.69	3.20	3.69	10.75	2.57	6.58	5.91
	Forward Best-First	1.76	3.24	**2.33**	**7.58**	**1.87**	**4.11**	**3.78**
	[88]	5.69	6.45	42.94	89.68	19.76	20.06	46.83
MAPE	None	**0.03**	2.91	1.50	5.24	0.96	3.11	1.77
	Forward Greedy-Stepwise	**0.03**	3.14	0.80	2.16	1.05	2.77	1.49
	Forward Best-First	0.04	3.22	**0.41**	**0.85**	**0.51**	**2.30**	**0.90**
	[88]	0.24	**2.79**	5.66	9.91	2.67	2.91	6.25

	Event log:	BPIC17	Hospital Bill	Traffic Fine	Credit Requirement	BPIC18	BPIC19	
	Attribute Selection Method							
Accuracy	None	**0.46**	**0.58**	**0.75**	**0.69**	**0.92**	**0.84**	
	Forward Greedy-Stepwise	0.44	0.57	**0.75**	0.68	0.90	0.82	
	Forward Best-First	0.44	0.52	0.73	**0.69**	0.88	0.80	
	[88]	0.17	0.35	0.50	0.44	0.64	0.57	
MAE	None	**3.08**	**21.30**	**24.69**	**0.21**	**4.92**	**6.60**	
	Forward Greedy-Stepwise	3.11	21.50	27.99	0.22	10.87	9.07	
	Forward Best-First	4.64	26.19	31.81	0.22	15.36	10.72	
	[88]	6.28	52.33	148.36	0.36	46.38	22.66	
MAPE	None	4.26	**3.41**	**0.01**	**0.00**	4.08	0.07	
	Forward Greedy-Stepwise	4.55	3.68	**0.01**	**0.00**	**2.23**	**0.05**	
	Forward Best-First	4.94	9.90	0.03	**0.00**	5.65	0.18	
	[88]	**0.40**	7.00	**0.01**	227.12	318.41	239.15	

Table 6.5: Time consumed for the different selection methods - time in the format of hh:mm:ss

	Event log:	BPIC12w	BPIC13	BPIC15_1	BPIC15_2	BPIC15_3	BPIC15_4	BPIC15_5
	Attribute Selection Method							
Time Consumed	None	**0:00:53**	**0:01:32**	7:25:13	6:13:23	13:00:44	5:50:27	6:17:30
	Forward Greedy-Stepwise	0:01:00	0:01:35	5:41:47	3:58:08	6:02:28	4:34:55	5:21:56
	Forward Best-First	0:01:04	0:01:40	**4:11:09**	**3:12:43**	**5:20:18**	**4:26:23**	**5:10:14**

	Event log:	BPIC17	Hospital Bill	Traffic Fine	Credit Requirement	BPIC18	BPIC19	
	Attribute Selection Method							
Time Consumed	None	0:00:20	1:53:16	**4:15:07**	**0:03:19**	11:11:34	2:07:02	
	Forward Greedy-Stepwise	00.00.39	2:02:06	4:15:58	0:03:28	10:43:09	3:39:21	
	Forward Best-First	**00.00.16**	**1:51:22**	4:17:04	0:03:50	**5:49:47**	**3:39:08**	

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

7.1 Conclusions

In this Ph.D. dissertation, we have addressed one of the current challenges in process mining enhancement: the prediction of remaining times in business processes. Accurate predictions of the remaining time, defined as the required time for an instance process to finish, are critical in many systems for organisations to be able to establish *a priori* requirements for the optimal management of resources or for improving the quality of the services that they provide. This Ph.D. dissertation has justified what we hypothesised at the beginning of this study from two different perspectives.

- **The structural information in the traces of a Business Process provides relevant information for achieving accurate remaining time predictions**. The features we collected from the event log provide our work with a more accurate result than other works in the literature. The structural information in the traces we collected helped us in this work not only by achieving accurate remaining time predictions, but by also letting us apply our model to any event log regardless of its case attributes such as ID, Resources, etc.

- **There are no models in the literature which consider the structural information in their estimations**. We build our work as we believe that this valuable information extracted from the event logs will enhance the prediction of the remaining time of a process.

The main conclusions can be summarised as the following:

First, in Chapter 4 we proposed 'A Novel Time Prediction Model Based on Structural Information From the Traces'. Our approach consists of two perspectives: firstly, we define a number of attributes that are evaluated from the process traces and capture quantitative and structural information about them. Secondly, a linear regression model is used for remaining time prediction using these attributes. The attributes are added to the well-known annotated transition system (ATS, [88]), thus producing a new Extended ATS which takes into account structural information of the traces. We have validated our model using ten real-life event logs of BPI Challenges 2012w, 2013, 2015, 2017 and Hospital Bill, and Traffic Fine [1]. We compared our approach to the baseline ATS model described in [88] only because this was our first model, and in the first instance we wanted to confirm that our model achieves our hypothesis. Results concluded that our model on average enhances accuracy by about 50%, and 35 days in MAE, and 2% in MAPE.

Second, in Chapter 5 'An Enhanced Model Including Lists Partitioning' was proposed, in which we developed the model we presented in Chapter 4 to improve the results so that we can compare it with other models in the literature (a non ATS-based model). As described in Chapter 4, each of the lists related to the state contains all the data (predictors or independent variables) needed for performing the remaining time estimation. Therefore, we have a single dataset associated with each state. Then, when applying a linear regression technique to each of these datasets, poor estimations are produced since, usually, traces in the dataset have great variability in terms of size, number of activities and execution times. This is the usual general scenario for real business process data, such as administrative procedures or applications, industrial incidents management or processes in a hospital or other big organisations/institutions [92, 94]. In many cases, the remaining time values range is vast (e.g. from a few seconds to a hundred thousand seconds) even for traces that are very similar or even identical.

In this model, our partitioning method consists of building partitions that contain partial traces with similar remaining times. This model allows for a number of partitions to be produced, hence, applying the linear regression to these partitions will produce a regression expression for each partition. Therefore, the estimation of the remaining time will be more accurate for the new partial trace that is related to the corresponded partition.

We have validated our model using ten real-life event logs of BPI Challenges 2012w, 2013, 2015, 2017, Hospital Bill and Traffic Fine [1]. The results concluded in two different comparison are as follows:

- We compared our approach to the baseline ATS model described in [88] using different thresholds. Results conclude that our model in the accuracy average is about 0.81 against 0.40 for the baseline [88]. In MAE, our model has an average of 6.71 days against 39.1 days.

- We compared our approach to different models in the literature, our model performing with all other models as described in Table 5.6.

Third, in Chapter 6 An ”Addressing Scalability of the Model” was proposed. After we introduced the partitioning technique to enhance the results we have in the basic model in Chapter 4, we proved how the prediction accuracy becomes more accurate for different threshold values. However, our model involves a high number of activities, which would also mean that the number of attributes could also be high. Consequently, the number of operations related to the partitioning stage would also increase. Therefore, in order to keep our approach general, we introduced attribute selection methods that reduce the number of attributes instances initially considered.

Two attribute selection methods were introduced. The first method is *Forward Greedy Stepwise* [103], which performs a greedy forward or backward search using the space of attribute subsets. The second method is *Forward Best-First* [103], which searches the scope of attribute sets by greedy *hill-climbing* expanded with a backtracking tactic.

We have validated our model using ten real-life event logs of BPI Challenges 2012w, 2013, 2015, 2017, 2018, 2019, Hospital Bill, and Traffic Fine [1]. Comparison results for the Accuracy, MAE and MAPE are the following:

- In the accuracy comparison, *Forward Greedy Stepwise* and *Forward Best-First* has an improvement of 24%.

- In the MAE comparison, *Forward Greedy Stepwise* and *Forward Best-First* has an improvement of 26 days.

- In the MAPE comparison, *Forward Greedy Stepwise* and *Forward Best-First* has an improvement of 97%.

In this model, we not only improved the result compared to the baseline [88], but also solved the problem with the scalability of the model.

7.2 Future Work

Here we present a list of various future research directions that can be pursued in order to continue the work started in this thesis, comprising:

- The partitioning method could be improved to adapt better to other more complex processes. Therefore, other partitioning methods could be proposed and compared with the current model, which could further enhance the good remaining time estimation results we have obtained with the current proposal.

- The proposed attributes may not be able to capture all the structural richness in some scenarios. In this regard, new attributes definitions could be considered.

- The use of regression techniques other than linear regression could be considered.

Bibliography

[1] 4TU.ResearchData. Real-life event logs. Available at: `https://data.4tu.nl/repository/collection\protect\leavevmode@ifvmode\kern+.2222em\relaxevent_logs_real`.

[2] Ahmad Aburomman, Manuel Lama, and Alberto Bugarín. Estimating remaining time in business processes using structural attributes of the traces. In *Computational Intelligence and Mathematics for tackling complex problems 2, 2019, Switzerland*, 2019.

[3] Ahmad Aburomman, Manuel Lama, and Alberto Bugarín. A vector-based classification approach for remaining time prediction in business processes. *IEEE Access*, 7:128198–128212, 2019.

[4] Arya Adriansyah, Boudewijn F. van Dongen, and Wil M. P. van der Aalst. Conformance checking using cost-based fitness analysis. In *IEEE 15th International Enterprise Distributed Object Computing Conference (EDOC 2011), Helsinki, Finland, August 29 - September 2, 2011*, pages 55–64. IEEE Computer Society, 2011.

[5] Rakesh Agrawal, Dimitrios Gunopulos, and Frank Leymann. Mining process models from workflow logs. In Hans-Jörg Schek, Fèlix Saltor, Isidro Ramos, and Gustavo Alonso, editors, *6th International Conference on Extending Database Technology (EDBT 1998), Valencia, Spain, March 23-27, 1998, Proceedings*, volume 1377, pages 469–483. Springer, Berlin, Heidelberg, 1998.

[6] Antonio Bevacqua, Marco Carnuccio, Francesco Folino, Massimo Guarascio, and Luigi Pontieri. A data-adaptive trace abstraction approach to the prediction of business process performances. In Slimane Hammoudi, Leszek A. Maciaszek, José Cordeiro,

and Jan L. G. Dietz, editors, *15th International Conference on Enterprise Information Systems (ICEIS 2013), Angers, France, 4-7 July, 2013*, volume 3, pages 56–65. INSTICC, SciTePress, 2013.

[7] Hendrik Blockeel. Top-down induction of first order logical decision trees. *AI Commun.*, 12(1-2):119–120, 1999.

[8] Alfredo Bolt and Marcos Sepúlveda. Process remaining time prediction using query catalogs. In *International Workshops on Business Process Management (BPM 2013), Beijing, China, August 26, 2013, Revised Papers*, pages 54–65, 2013.

[9] Joos C. A. M. Buijs, Marcello La Rosa, Hajo A. Reijers, Boudewijn F. van Dongen, and Wil M. P. van der Aalst. Improving business process models using observed behavior. In Philippe Cudré-Mauroux, Paolo Ceravolo, and Dragan Gasevic, editors, *International Symposium on Data-Driven Process Discovery and Analysis (SIMPDA 2012), Campione d'Italia, Italy, June 18-20, 2012, Revised Selected Papers*, volume 162, pages 44–59. Springer Berlin Heidelberg, 2012.

[10] Ramon Casadesus-Masanell and Joan Enric Ricart. From strategy to business models and onto tactics. *Long Range Planning*, 43(2):195 – 215, 2010.

[11] Michelangelo Ceci and Annalisa Appice. Spatial associative classification: propositional vs structural approach. *J. Intell. Inf. Syst.*, 27(3):191–213, 2006.

[12] 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), Bled, Slovenia, October 8-10, 2014*, pages 49–61, 2014.

[13] Eugenio Cesario, Francesco Folino, Massimo Guarascio, and Luigi Pontieri. A cloud-based prediction framework for analyzing business process performances. In *5th International Cross-Domain Conference on Availability, Reliability, and Security in Information Systems (CD-ARES 2016), Salzburg, Austria, August 31 - September 2, 2016, Proceedings*, volume 9817, pages 63–80. Springer, Cham, 2016.

[14] Eugenio Cesario and Domenico Talia. Distributed data mining patterns and services: an architecture and experiments. *Concurrency and Computation: Practice and Experience*, 24(15):1751–1774, 2012.

[15] James F Chang. *Business process management systems: strategy and implementation*. Auerbach Publications NY, USA, 2016.

[16] George Choueiry. Understand Forward and Backward Stepwise Regression. Available at: `https://quantifyinghealth.com/stepwise-selection`, 2020.

[17] Edmund M. Clarke, Orna Grumberg, and Doron A. Peled. *Model checking*. MIT Press, Cambridge, MA, USA, 2001.

[18] Sophie Cluet. Review - mining association rules between sets of items in large databases. *ACM SIGMOD Digital Review*, 1, 1999.

[19] Raffaele Conforti, Massimiliano de Leoni, Marcello La Rosa, and Wil M. P. van der Aalst. Supporting risk-informed decisions during business process execution. In Camille Salinesi, Moira C. Norrie, and Oscar Pastor, editors, *25th International Conference on Advanced Information Systems Engineering (CAiSE 2013), Valencia, Spain, June 17-21, 2013. Proceedings*, volume 7908, pages 116–132. Springer, Berlin, Heidelberg, 2013.

[20] Massimiliano de Leoni and Wil M. P. van der Aalst. Data-aware process mining: discovering decisions in processes using alignments. In *28th Annual ACM Symposium on Applied Computing (SAC 2013), Coimbra, Portugal, March 18-22, 2013*, pages 1454–1461. ACM, 2013.

[21] Ana Karla A. de Medeiros, A. J. M. M. Weijters, and Wil M. P. van der Aalst. Genetic process mining: an experimental evaluation. *Data Min. Knowl. Discov.*, 14(2):245–304, 2007.

[22] Brian Denton, James Viapiano, and Andrea Vogl. Optimization of surgery sequencing and scheduling decisions under uncertainty. *Health Care Management Science*, 10(1):13–24, Feb 2007.

[23] Prabhakar M. Dixit, Joos C. A. M. Buijs, Wil M. P. van der Aalst, Bart Hompes, and Hans Buurman. Enhancing process mining results using domain knowledge. In Paolo Ceravolo and Stefanie Rinderle-Ma, editors, *5th International Symposium on Data-driven Process Discovery and Analysis (SIMPDA 2015), Vienna, Austria, December 9-11, 2015*, volume 1527, pages 79–94. CEUR-WS.org, 2015.

[24] Marlon Dumas. From models to data and back: the journey of the bpm discipline and the tangled road to bpm 2020. In *13th International Conference on Business Process Management*, number 9253 in 2. Springer, Heidelberg, 2015.

[25] Joerg Evermann, Jana-Rebecca Rehse, and Peter Fettke. A deep learning approach for predicting process behaviour at runtime. In Marlon Dumas and Marcelo Fantinato, editors, *International Workshops on Business Process Management (BPM 2016), Rio de Janeiro, Brazil, September 19, 2016*, volume 281, pages 327–338. Springer, Cham, 2016.

[26] Dirk Fahland and Wil M. P. van der Aalst. Repairing process models to reflect reality. In Alistair P. Barros, Avigdor Gal, and Ekkart Kindler, editors, *10th International Conference on Business Process Management (BPM 2012), Tallinn, Estonia, September 3-6, 2012. Proceedings*, volume 7481, pages 229–245. Springer, Berlin, Heidelberg, 2012.

[27] Francesco Folino, Massimo Guarascio, and Luigi Pontieri. Context-aware predictions on business processes: An ensemble-based solution. In *First International Workshop on New Frontiers in Mining Complex Patterns (NFMCP 2012), Held in Conjunction with ECML/PKDD 2012, Bristol, UK, September 24, 2012, Revised Selected Papers*, volume 7765 of *Lecture Notes in Computer Science*, pages 215–229. Springer, 2012.

[28] Francesco Folino, Massimo Guarascio, and Luigi Pontieri. Discovering context-aware models for predicting business process performances. In Robert Meersman, Hervé Panetto, Tharam S. Dillon, Stefanie Rinderle-Ma, Peter Dadam, Xiaofang Zhou, Siani Pearson, Alois Ferscha, Sonia Bergamaschi, and Isabel F. Cruz, editors, *Confederated International Conferences On the Move to Meaningful Internet Systems (OTM 2012), Rome, Italy, September 10-14, 2012*, volume 7565, pages 287–304. Springer, Berlin, Heidelberg, 2012.

[29] Ian T. Foster. Globus toolkit version 4: Software for service-oriented systems. *J. Comput. Sci. Technol.*, 21(4):513–520, 2006.

[30] Chiara Di Francescomarino, Marlon Dumas, Marco Federici, Chiara Ghidini, Fabrizio Maria Maggi, and Williams Rizzi. Predictive business process monitoring framework with hyperparameter optimization. In Selmin Nurcan, Pnina Soffer, Marko Bajec,

and Johann Eder, editors, *28th International Conference on Advanced Information Systems Engineering (CAiSE 2016), Ljubljana, Slovenia, June 13-17, 2016. Proceedings*, volume 9694, pages 361–376. Springer International Publishing, 2016.

[31] Chiara Di Francescomarino, Marlon Dumas, Fabrizio Maria Maggi, and Irene Teinemaa. Clustering-based predictive process monitoring. *IEEE Trans. Services Computing*, 12(6):896–909, 2019.

[32] Chiara Di Francescomarino, Chiara Ghidini, Fabrizio Maria Maggi, and Fredrik Milani. Predictive process monitoring methods: Which one suits me best? *CoRR*, abs/1804.02422, 2018.

[33] George M Giaglis, Ioannis Minis, Antonios Tatarakis, and Vasileios Zeimpekis. Minimizing logistics risk through real-time vehicle routing and mobile technologies: Research to date and future trends. *International Journal of Physical Distribution & Logistics Management*, 34(9):749–764, 2004.

[34] E. Mark Gold. Complexity of automaton identification from given data. *Information and Control*, 37(3):302–320, 1978.

[35] Alex Graves, Abdel-rahman Mohamed, and Geoffrey E. Hinton. Speech recognition with deep recurrent neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2013), Vancouver, BC, Canada, May 26-31, 2013*, pages 6645–6649. IEEE, 2013.

[36] Peter Gyngell. Reengineering the corporation: A manifesto for business revolution. *J. Strategic Inf. Sys.*, 3(4):339–345, 1994.

[37] David Harel. *Come, let's play - scenario-based programming using LSCs and the play-engine, 1st edition*. Springer-Verlag Berlin Heidelberg, 2003.

[38] Trevor Hastie, Robert Tibshirani, and Jerome H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, 2nd Edition*. Springer-Verlag New York, 2009.

[39] Joachim Herbst and Dimitris Karagiannis. Integrating machine learning and workflow management to support acquisition and adaptation of workflow models. *Int. Syst. in Accounting, Finance and Management*, 9(2):67–92, 2000.

[40] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[41] Vojtech Huser. Process mining: Discovery, conformance and enhancement of business processes. *Journal of Biomedical Informatics*, 45(5):1018–1019, 2012.

[42] Marta Indulska, Peter F. Green, Jan Recker, and Michael Rosemann. Business process modeling: Perceived benefits. In Alberto H. F. Laender, Silvana Castano, Umeshwar Dayal, Fabio Casati, and Jos'e Palazzo Moreira de Oliveira, editors, *28th International Conference on Conceptual Modeling (ER 2009), Gramado, Brazil, November 9-12, 2009. Proceedings*, volume 5829, pages 458–471. Springer, Berlin, Heidelberg, 2009.

[43] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning with Applications in R*. Springer New York, 2013.

[44] Anthony D Josep, Randy Katz, Andy Konwinski, LEE Gunho, David Patterson, and Ariel Rabkin. A view of cloud computing. *Communications of the ACM*, 53(4), 2010.

[45] Laura J Kornish and Jeremy Hutchison-Krupat. Research on idea generation and selection: Implications for management of technology. *Production and Operations Management*, 26(4):633–651, 2017.

[46] Huawen Liu, Jigui Sun, Lei Liu, and Huijie Zhang. Feature selection with dynamic mutual information. *Pattern Recognition*, 42(7):1330–1339, 2009.

[47] Ruopeng Lu and Shazia Wasim Sadiq. A survey of comparative business process modeling approaches. In Witold Abramowicz, editor, *10th International Conference on Business Information Systems (BIS 2007), Poznan, Poland, April 25-27, 2007, Proceedings*, volume 4439, pages 82–94. Springer, Berlin, Heidelberg, 2007.

[48] David Luckham. The power of events: An introduction to complex event processing in distributed enterprise systems. In Nick Bassiliades, Guido Governatori, and Adrian Paschke, editors, *International Workshop on Rules and Rule Markup Languages for the Semantic Web (RuleML 2008), Orlando, FL, USA, October 30-31, 2008. Proceedings*, volume 5321, page 3. Springer, Berlin, Heidelberg, 2008.

[49] Fabrizio Maria Maggi and Michael Westergaard. Designing software for operational decision support through coloured petri nets. *Enterprise IS*, 11(5):576–596, 2017.

[50] Felix Mannhardt. Hospital Billing - event log. Available at: `https://doi.org/10.4121/uuid:76c46b83-c930-4798-a1c9-4be94dfeb741`, 2016.

[51] Felix Mannhardt and Massimiliano de Leoni. Road Traffic Fine Management Process. Available at: `https://doi.org/10.4121/uuid:270fd440-1057-4fb9-89a9-b699b47990f5`, 2014.

[52] Selma Limam Mansar and Hajo A. Reijers. Best practices in business process redesign: use and impact. *Business Proc. Manag. Journal*, 13(2):193–213, 2007.

[53] Alfonso Eduardo Márquez-Chamorro, Manuel Resinas, and Antonio Ruiz-Cortés. Predictive monitoring of business processes: A survey. *IEEE Trans. Services Computing*, 11(6):962–977, 2018.

[54] Ayman Meidan, J. A. García-García, María José Escalona Cuaresma, and Isabel M. Ramos. A survey on business processes management suites. *Computer Standards & Interfaces*, 51:71–86, 2017.

[55] Jan Mendling, Hajo A. Reijers, and Wil M. P. van der Aalst. Seven process modeling guidelines (7PMG). *Information & Software Technology*, 52(2):127–136, 2010.

[56] Andreas Metzger, Philipp Leitner, Dragan Ivanovic, Eric Schmieders, Rod Franklin, Manuel Carro, Schahram Dustdar, and Klaus Pohl. Comparing and combining predictive business process monitoring techniques. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(2):276–290, 2015.

[57] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *26th Annual Conference on Neural Information Processing Systems (NIPS 2013), Lake Tahoe, Nevada, United States, December 5-8, 2013*, volume 2, pages 3111–3119. Curran Associates Inc., 2013.

[58] Germán Moltó and Vicente Hernández. On demand replication of wsrf-based grid services via cloud computing. In *9th International Meeting on High Performance Computing for Computational Science (VecPar 2010)*, 2010.

[59] Hamid R. Motahari Nezhad, Régis Saint-Paul, Fabio Casati, and Boualem Benatallah. Event correlation for process discovery from web service interaction logs. *VLDB J.*, 20(3):417–444, 2011.

[60] Kreshnik Musaraj, Tetsuya Yoshida, Florian Daniel, Mohand-Said Hacid, Fabio Casati, and Boualem Benatallah. Message correlation and web service protocol mining from inaccurate logs. In *IEEE International Conference on Web Services (ICWS 2010), Miami, Florida, USA, July 5-10, 2010*, pages 259–266. IEEE Computer Society, 2010.

[61] OMG. *Business Process Model and Notation BPMN.* Springer, Berlin, Heidelberg, 2011.

[62] OMG. Unified modeling language, infrastructure, v2.1.4. *Object Management Group*, 2011.

[63] Suraj Pandey, Surya Nepal, and Shiping Chen. A test-bed for the evaluation of business process prediction techniques. In Dimitrios Georgakopoulos and James B. D. Joshi, editors, *7th International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom 2011), Orlando, FL, USA, 15-18 October, 2011*, pages 382–391. ICST / IEEE, 2011.

[64] Anastasiia Pika, Wil M. P. van der Aalst, Colin J. Fidge, Arthur H. M. ter Hofstede, and Moe Thandar Wynn. Predicting deadline transgressions using event logs. In Marcello La Rosa and Pnina Soffer, editors, *International Workshops on Business Process Management(BPM 2012), Tallinn, Estonia, September 3, 2012*, volume 132, pages 211–216. Springer, Berlin, Heidelberg, 2012.

[65] Roger Pinkham. Applied nonparametric regression (wolfgang hardle). *SIAM Review*, 34(2):341–342, 1992.

[66] Mirko Polato, Alessandro Sperduti, Andrea Burattin, and Massimiliano de Leoni. Data-aware remaining time prediction of business process instances. In *International Joint Conference on Neural Networks, (IJCNN 2014), Beijing, China, July 6-11, 2014*, pages 816–823, 2014.

[67] Sonja Pravilovic, Annalisa Appice, and Donato Malerba. Process mining to forecast the future of running cases. In Annalisa Appice, Michelangelo Ceci, Corrado Loglisci,

Giuseppe Manco, Elio Masciari, and Zbigniew W. Ras, editors, *Second International Workshop on New Frontiers in Mining Complex Patterns (NFMCP 2013), Prague, Czech Republic, September 27, 2013*, volume 8399, pages 67–81. Springer, Cham, 2013.

[68] Jeff Racine and Qi Li. Nonparametric estimation of regression functions with both categorical and continuous data. *Journal of Econometrics*, 119(1):99–130, 2004.

[69] Aubrey J. Rembert, Amos Omokpo, Pietro Mazzoleni, and Richard Goodwin. Process discovery using prior knowledge. In Samik Basu, Cesare Pautasso, Liang Zhang, and Xiang Fu, editors, *11th International Conference on Service-Oriented Computing (ICSOC 2013), Berlin, Germany, December 2-5, 2013, Proceedings*, volume 8274, pages 328–342. Springer, Berlin, Heidelberg, 2013.

[70] Andreas Rogge-Solti and Mathias Weske. Prediction of remaining service execution time using stochastic petri nets with arbitrary firing delays. In Samik Basu, Cesare Pautasso, Liang Zhang, and Xiang Fu, editors, *11th International Conference on Service-Oriented Computing (ICSOC 2013), Berlin, Germany, December 2-5, 2013, Proceedings*, volume 8274, pages 389–403. Springer, Berlin, Heidelberg, 2013.

[71] Andreas Rogge-Solti and Mathias Weske. Prediction of business process durations using non-markovian stochastic petri nets. *Inf. Syst.*, 54:1–14, 2015.

[72] Anne Rozinat and Wil M. P. van der Aalst. Conformance testing: Measuring the fit and appropriateness of event logs and process models. In Christoph Bussler and Armin Haller, editors, *International Workshops on Business Process Management, (BPM 2005), Nancy, France, September 5, 2005*, volume 3812, pages 163–176. Springer, Berlin, Heidelberg, 2005.

[73] Anne Rozinat and Wil M. P. van der Aalst. Conformance checking of processes based on monitoring real behavior. *Inf. Syst.*, 33(1):64–95, 2008.

[74] Mari Sako. Business models for strategy and innovation. *Commun. ACM*, 55(7):22–24, 2012.

[75] Peter Sempolinski and Douglas Thain. A comparison and critique of eucalyptus, opennebula and nimbus. In *2ed International Conference on Cloud Computing (CloudCom*

2010), Indianapolis, Indiana, USA, November 30 - December 3, 2010, pages 417–426. IEEE Computer Society, 2010.

[76] Arik Senderovich, Chiara Di Francescomarino, Chiara Ghidini, Kerwin Jorbina, and Fabrizio Maria Maggi. Intra and inter-case features in predictive process monitoring: A tale of two dimensions. In *15th International Conference on Business Process Management (BPM 2017), Barcelona, Spain, September 10-15, 2017, Proceedings*, volume 10445, pages 306–323. Springer International Publishing, 2017.

[77] Ward Steeman. BPI Challenge 2013. Available at: `https://doi.org/10.4121/uuid:a7ce5c55-03a7-4583-b855-98b86e1a2b07`, 2014.

[78] Niek Tax, Ilya Verenich, Marcello La Rosa, and Marlon Dumas. Predictive business process monitoring with LSTM neural networks. In Eric Dubois and Klaus Pohl, editors, *29th International Conference on Advanced Information Systems Engineering (CAiSE 2017), Essen, Germany, June 12-16, 2017, Proceedings*, volume 10253, pages 477–492. Springer, Cham, 2017.

[79] Wil M. P. van der Aalst. Process mining: Overview and opportunities. *ACM Trans. Manage. Inf. Syst.*, 3(2), 2012.

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

[81] Wil M. P. van der Aalst. Process modeling and analysis. In *Process Mining, Data Science in Action*, pages 55–88. Springer, Berlin, Heidelberg, 2016.

[82] Wil M. P. van der Aalst, Arya Adriansyah, and Ana Karla Alves De Medeiros. Process mining manifesto. *Lecture Notes in Business Information Processing*, 99 LNBIP(PART 1):169–194, 2012.

[83] Wil M. P. van der Aalst and S. Dustdar. Process Mining Put into Context. *IEEE Internet Computing*, 16(1):82–86, 2012.

[84] Wil M. P. van der Aalst and Schahram Dustdar. Process mining put into context. *IEEE Internet Computing*, 16(1):82–86, 2012.

[85] Wil M. P. van der Aalst, Maja Pesic, and Minseok Song. Beyond process mining: From the past to present and future. In *22nd International Conference on Advanced*

Information Systems Engineering (CAiSE 2010), Hammamet, Tunisia, June 7-9, 2010, volume 6051 of *Lecture Notes in Computer Science*, pages 38–52. Springer, 2010.

[86] Wil M. P. van der Aalst, Marcello La Rosa, and Flávia Maria Santoro. Business process management - don't forget to improve the process! *Business & Information Systems Engineering*, 58(1):1–6, 2016.

[87] Wil M. P. van der Aalst, Vladimir A. Rubin, H. M. W. Verbeek, Boudewijn F. van Dongen, Ekkart Kindler, and Christian W. Günther. Process mining: a two-step approach to balance between underfitting and overfitting. *Software and System Modeling*, 9(1):87–111, 2010.

[88] 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.

[89] Wil M. P. van der Aalst, Arthur H. M. ter Hofstede, and Mathias Weske. Business process management: A survey. In Wil M. P. van der Aalst, Arthur H. M. ter Hofstede, and Mathias Weske, editors, *International Conference on Business Process Management (BPM 2003), Eindhoven, The Netherlands, June 26-27, 2003, Proceedings*, volume 2678, pages 1–12. Springer, Berlin, Heidelberg, 2003.

[90] Wil M. P. van der Aalst, Boudewijn F. van Dongen, Christian W. Günther, R. S. Mans, Ana Karla Alves de Medeiros, Anne Rozinat, Vladimir A. Rubin, Minseok Song, H. M. W. Verbeek, and A. J. M. M. Weijters. Prom 4.0: Comprehensive support for *Real* process analysis. In *28th International Conference on Applications and Theory of Petri Nets and Other Models of Concurrency (ICATPN 2007), Siedlce, Poland, June 25-29, 2007, Proceedings*, volume 4546, pages 484–494. Springer, Berlin, Heidelberg, 2007.

[91] Wil M. P. van der Aalst, Ton Weijters, and Laura Maruster. Workflow mining: Discovering process models from event logs. *IEEE Trans. Knowl. Data Eng.*, 16(9):1128–1142, 2004.

[92] Boudewijn F. van Dongen. BPI Challenge 2012. Available at: `http://data.4tu.nl/repository/uuid:3926db30-f712-4394-aebc-75976070e91f`, 2012.

[93] Boudewijn F. van Dongen. BPI Challenge 2015. Available at: `https://doi.org/10.4121/uuid:31a308ef-c844-48da-948c-305d167a0ec1`, 2015.

[94] Boudewijn F. van Dongen. BPI Challenge 2017. Available at: `https://doi.org/10.4121/uuid:5f3067df-f10b-45da-b98b-86ae4c7a310b`, 2017.

[95] Boudewijn F. van Dongen, R. A. Crooy, and Wil M. P. van der Aalst. Cycle time prediction: When will this case finally be finished? In *Confederated International Conferences On the Move to Meaningful Internet Systems (OTM 2008), Monterrey, Mexico, November 9-14, 2008, Part I*, pages 319–336, 2008.

[96] Rob J. van Glabbeek and W. P. Weijland. Branching time and abstraction in bisimulation semantics. *J. ACM*, 43(3):555–600, 1996.

[97] Ilya Verenich, Marlon Dumas, Marcello La Rosa, Fabrizio Maria Maggi, and Irene Teinemaa. Survey and cross-benchmark comparison of remaining time prediction methods in business process monitoring. *ACM Transactions on Intelligent Systems and Technology (TIST 2019)*, 10(4):34:1–34:34, 2019.

[98] Ilya Verenich, Hoang Nguyen, Marcello La Rosa, and Marlon Dumas. White-box prediction of process performance indicators via flow analysis. In *International Conference on the Software and Systems Process (ICSSP 2017), Paris, France, July 5-7, 2017*, pages 85–94. ACM New York, NY, USA, 2017.

[99] Jan vom Brocke, Sarah Zelt, and Theresa Schmiedel. On the role of context in business process management. *International Journal of Information Management*, 36(3):486–495, 2016.

[100] Mathias Weske. *Business Process Management - Concepts, Languages, Architectures, 2nd Edition*. Springer, Berlin, Heidelberg, 2012.

[101] Michael Westergaard and Fabrizio Maria Maggi. Modeling and verification of a protocol for operational support using coloured petri nets. In Lars Michael Kristensen and Laure Petrucci, editors, *32nd International Conference on Applications and Theory of Petri Nets (PETRI NETS 2011), Newcastle, UK, June 20-24, 2011. Proceedings*, volume 6709, pages 169–188. Springer, Berlin, Heidelberg, 2011.

[102] Bernd W Wirtz, Adriano Pistoia, Sebastian Ullrich, and Vincent Göttel. Business models: Origin, development and future research perspectives. *Long range planning*, 49(1):36–54, 2016.

[103] Ian H. Witten, Eibe Frank, and Mark A. Hall. *Data mining: practical machine learning tools and techniques, 3rd Edition*. Morgan Kaufmann, Elsevier, 2011.

[104] Xi Zhao, Wei Deng, and Yong Shi. Feature selection with attributes clustering by maximal information coefficient. In Yong Shi, Youmin Xi, Peter Wolcott, Yingjie Tian, Jianping Li, Daniel Berg, Zhengxin Chen, Enrique Herrera-Viedma, Gang Kou, Heeseok Lee, Yi Peng, and Lean Yu, editors, *1st International Conference on Information Technology and Quantitative Management, (ITQM 2013, Dushu Lake Hotel, Sushou, China, 16-18 May, 2013*, volume 17, pages 70–79. Elsevier B.V., 2013.

[105] Michael zur Muehlen and Robert Shapiro. Business process analytics. In Jan vom Brocke and Michael Rosemann, editors, *Handbook on Business Process Management 2, Strategic Alignment, Governance, People and Culture, 2nd Ed*, pages 243–263. Springer-Verlag Berlin Heidelberg, 2015.

List of Figures

List of Tables