UNIVERSIDAD DE SEVILLA

DOCTORAL THESIS

On the enhancement of Big Data Pipelines through Data Preparation, Data Quality, and the distribution of Optimisation Problems

---

Author:
Álvaro Valencia Parra

Advised by:
Dr. Mª Teresa Gómez López
Dr. Angel J. Varela Vaca

A thesis submitted in fulfillment of the requirements for the degree of Doctor in Computer Engineering

in the

IDEA Research Group
Department of Computer Languages and Systems
Escuela Técnica Superior de Ingeniería Informática

June, 2022

UNIVERSIDAD DE SEVILLA

TESIS DOCTORAL

On the enhancement of Big Data Pipelines through Data Preparation, Data Quality, and the distribution of Optimisation Problems

---

Autor:
Álvaro Valencia Parra

Dirigida por:
Dr. Mª Teresa Gómez López
Dr. Angel J. Varela Vaca

Memoria que presenta para optar al título de
Doctor en Informática

realizada en

IDEA Research Group
Departamento de Lenguajes y Sistemas Informáticos
Escuela Técnica Superior de Ingeniería Informática

Junio de 2022

A mis padres.

UNIVERSIDAD DE SEVILLA

Abstract

Escuela Técnica Superior de Ingeniería Informática
Departamento de Lenguajes y Sistemas Informáticos

Doctor in Computer Engineering

On the enhancement of Big Data Pipelines through Data Preparation, Data Quality, and the distribution of Optimisation Problems

by Álvaro Valencia Parra

Nowadays, data are fundamental for companies, providing operational support by facilitating daily transactions. Data has also become the cornerstone of strategic decision-making processes in businesses. For this purpose, there are numerous techniques that allow to extract knowledge and value from data. For example, optimisation algorithms excel at supporting decision-making processes to improve the use of resources, time and costs in the organisation. In the current industrial context, organisations usually rely on business processes to orchestrate their daily activities while collecting large amounts of information from heterogeneous sources. Therefore, the support of Big Data technologies (which are based on distributed environments) is required given the volume, variety and speed of data. Then, in order to extract value from the data, a set of techniques or activities is applied in an orderly way and at different stages. This set of techniques or activities, which facilitate the acquisition, preparation, and analysis of data, is known in the literature as Big Data pipelines.

In this thesis, the improvement of three stages of the Big Data pipelines is tackled: Data Preparation, Data Quality assessment, and Data Analysis. These improvements can be addressed from an individual perspective, by focussing on each stage, or from a more complex and global perspective, implying the coordination of these stages to create data workflows.

The first stage to improve is the Data Preparation by supporting the preparation of data with complex structures (i.e., data with various levels of nested structures, such as arrays). Shortcomings have been found in the literature and current technologies for transforming complex data in a simple way. Therefore, this thesis aims to improve the Data Preparation stage through Domain-Specific Languages (DSLs). Specifically, two DSLs are proposed for different use cases. While one of them is a general-purpose Data Transformation language, the other is a DSL aimed at extracting event logs in a standard format for process mining algorithms.

The second area for improvement is about the assessment of Data Quality. Depending on the type of Data Analysis algorithm, poor-quality data can seriously skew the results. A clear example are optimisation algorithms. If the data are not sufficiently accurate and complete,

the search space can be severely affected. Therefore, this thesis formulates a methodology for modelling Data Quality rules adjusted to the context of use, as well as a tool that facilitates the automation of their assessment. This allows to discard the data that do not meet the quality criteria defined by the organisation. In addition, the proposal includes a framework that helps to select actions to improve the usability of the data.

The third and last proposal involves the Data Analysis stage. In this case, this thesis faces the challenge of supporting the use of optimisation problems in Big Data pipelines. There is a lack of methodological solutions that allow computing exhaustive optimisation problems in distributed environments (i.e., those optimisation problems that guarantee the finding of an optimal solution by exploring the whole search space). The resolution of this type of problem in the Big Data context is computationally complex, and can be NP-complete. This is caused by two different factors. On the one hand, the search space can increase significantly as the amount of data to be processed by the optimisation algorithms increases. This challenge is addressed through a technique to generate and group problems with distributed data. On the other hand, processing optimisation problems with complex models and large search spaces in distributed environments is not trivial. Therefore, a proposal is presented for a particular case in this type of scenario.

As a result, this thesis develops methodologies that have been published in scientific journals and conferences. The methodologies have been implemented in software tools that are integrated with the Apache Spark data processing engine. The solutions have been validated through tests and use cases with real datasets.

UNIVERSIDAD DE SEVILLA

Resumen

Escuela Técnica Superior de Ingeniería Informática
Departamento de Lenguajes y Sistemas Informáticos

Doctor en Informática

On the enhancement of Big Data Pipelines through Data Preparation, Data Quality, and the distribution of Optimisation Problems

por Álvaro Valencia Parra

Hoy en día los datos se han consolidado como una pieza fundamental en las empresas, ya que, además de facilitar las transacciones del día a día a nivel operacional, ayudan a mejorar la toma de decisiones estratégicas en el negocio. Para ello, existen diversas técnicas que permiten extraer conocimiento y valor de los datos. Un ejemplo son los algoritmos de optimización, los cuales permiten utilizar los datos para dar soporte a la toma de decisiones con el fin de mejorar el uso de los recursos, tiempo y costes en todos los ámbitos de una organización. En el contexto industrial actual, las organizaciones frecuentemente se apoyan en procesos de negocio con los que orquestan sus actividades diarias al mismo tiempo que recolectan grandes cantidades de información de orígenes heterogéneos. Por ello, necesitan el soporte de tecnologías Big Data dado el volumen, la variedad y la velocidad de los datos. El procesamiento mediante técnicas Big Data se lleva a cabo en entornos distribuidos, en los que, según el caso de uso, se aplican una serie de métodos para extraer valor de los datos. Este conjunto de técnicas o actividades, aplicadas de una forma ordenada y en distintas etapas, se denomina Big Data *pipeline* y facilita la adquisición de los datos, la preparación, y su análisis.

Se han detectado aspectos a mejorar en las fases de preparación, valoración de la calidad, y análisis de datos. Estas se pueden abordar desde una perspectiva individual, enfocada en cada fase, o desde un nivel más complejo y global dentro de un workflow de datos. En esta tesis, abordamos la mejora de cada una de estas partes de distintas formas.

En primer lugar, esta tesis se enfoca en la preparación de datos con estructuras complejas, es decir, datos con estructuras anidadas a distintos niveles. Se han encontrado carencias en la literatura y en las tecnologías actuales para transformarlos de manera sencilla. Es por eso por lo que esta tesis se propone perfeccionar la preparación de datos complejos mediante Lenguajes Específicos de Dominio (*DSL*, por sus siglas en inglés). En concreto, se presentan dos, según el caso de uso: uno de propósito general, y otro orientado a la extracción de logs de eventos en un formato estándar para algoritmos de minería de procesos.

La segunda área de mejora está relacionada con la valoración de la calidad de los datos. Según el tipo de algoritmo de análisis que se utilice, los datos de mala calidad pueden llegar a

desvirtuar gravemente los resultados. Un claro ejemplo son los algoritmos de optimización. Si los datos no son lo suficientemente precisos y completos, el espacio de búsqueda se puede ver gravemente afectado. Por lo tanto, esta tesis formula una metodología para modelar reglas de calidad de datos ajustadas al contexto de uso, así como una herramienta que facilita la automatización de su valoración. Esto permite descartar aquellos que no cumplan con los criterios de calidad definidos por la organización. Además, la propuesta incluye un *framework* que ayuda en la selección de acciones para mejorar la usabilidad de los datos.

La tercera y última propuesta se enmarca en la fase del análisis de datos. Para ello, esta tesis afronta el reto de dar soporte al uso de problemas de optimización en Big Data *pipelines*. En particular, se ha detectado una falta de soluciones metodológicas que permitan computar problemas de optimización exhaustivos en entornos distribuidos, es decir, aquellos que requieren del descubrimiento de la solución óptima dentro de un espacio de búsqueda. La resolución de este tipo de problemas en el contexto Big Data es computacionalmente compleja, pudiendo llegar a ser de tipo NP-completo. Esto se debe a que, por una parte, el espacio de búsqueda puede aumentar de forma significativa a medida que la cantidad de datos que deben procesar los algoritmos de optimización se incremente. Este desafío es abordado a través de una técnica para generar y agrupar problemas con datos distribuidos. Por otra parte, el procesamiento de problemas de optimización con modelos complejos y grandes espacios de búsqueda en entornos distribuidos no es trivial. Por consiguiente, se presenta una propuesta para un caso particular en este tipo de escenarios.

Como resultado, se han desarrollado metodologías que han sido publicadas en artículos de investigación. Estas se han implementado en herramientas de software que se integran con el motor de procesamiento de datos masivos Apache Spark. Además, han sido validadas mediante pruebas y casos de uso con datos reales.

Acknowledgements

Firstly, I would like to express my deepest gratitude to my advisors, Mayte Gómez and Ángel Varela, for motivating me to start this project and guiding and supporting me throughout this journey. Thank you very much for your support and understanding in the most difficult moments, for all the hours you have dedicated to me, and for everything I have been able to learn thanks to your dedication.

I also express my gratitude to all the researchers with whom I have collaborated. Prof. Paolo Ceravolo, Prof. Ismael Caballero, Prof. Luisa Parody, Prof. Josep Carmona and Prof. Robin Bergenthum, among others. It has been a pleasure working and learning with you. I would also like to extend my gratitude to the members of the IDEA research group for always being willing to help me and for all the advice they have given me during this period.

I would like to mention the Department of Computer Languages and Systems, where I have had the pleasure of collaborating in teaching, and the University of Seville, for financing my predoctoral contract *VIPPIT-2019-II.2* in the framework of *Programa de Formación Predoctoral*.

Contents

Chapter 1

Introduction

This chapter contextualises this doctoral thesis, describing its context and motivation, introducing all the concepts necessary to understand the problems to be addressed and describing the set of challenges and objectives that are faced. This Chapter is structured as follows. Section 1.1 contextualises and motivates this thesis. Next, Section 1.2 provides a background on the main concepts that are closely related to this thesis. Afterwards, Section 1.3 describes our objectives. Finally, Section 1.4.1 presents the research methodology that has been followed.

1.1 Context and Motivation

1.1.1 Context

The technological advances in Industry are promoting changes towards complex business processes, leading to a paradigm in which data is the cornerstone of business operations and strategy. Traditionally, data played a relevant role within organisations. In the late 1970s, information systems emerged, and these provided operational support to companies through transaction processing systems [61]. In that scenario, the database management systems were responsible for storing operational data. Since then, operational data has supported strategic and tactical levels through analytical operations based on descriptive statistics [69]. Since the appearance of these systems capable of storing, processing, and analysing transactions at the operational level of companies, the importance of data in business processes has increased. For example, data can be employed to support decision-making processes through optimisation algorithms, which help companies use data to optimise resources, time, and costs within business processes.

As data became more dominant, companies sought to store as much data as possible. The current industrial context seeks to collect as much data as possible to increase the potential value extracted from it. The current trend [4] is the integration of data from multiple sources (e.g., data collected by sensors on manufacturing lines or third-party data, such as data from business partners, providers, clients, and social networks). All of these data, as a whole, are a potential source of value for companies. If data are properly processed and analysed, it can be convenient for optimising business processes, strengthening relationships with customers and suppliers, improving product and service quality, improving customer engagement, and developing new business or investment opportunities.

Although this scenario brings great opportunities, several challenges arise. The main challenges are related to the nature of this type of data, which follows three characteristics: There are large amounts of data (volume) generated at high speed (velocity) and showing high heterogeneity (variety). These challenges promoted the emergence of the Big Data paradigm to address these limitations. This paradigm is intended to capture, store, manage, and analyse data [54] that meet the characteristics of volume, velocity, and variety. These characteristics are formally described as *the three dimensions of Big Data* [52, 53]. Other authors [53, 2, 58] included an additional dimension: Veracity. Following these authors, this dimension is related to the uncertainty and unreliability that are implicit in certain data sources, motivated by the lack of quality of the data, manifested as accuracy, completeness, consistency, latency, ambiguity, and availability issues.

Each dimension of Big Data entails several challenges that must be addressed [8]. Focussing on the variety of data, note that processing heterogeneous data is not trivial. In Big Data scenarios, data with different semantics and structures (i.e., structured data with different attributes, or unstructured data such as plain text, logs, images, audio, videos [63, 53]) must be integrated. On the other hand, if we focus on the dimension of veracity, the challenges that arise are related to the quality, reproducibility, and security [91, 28] of the data. The higher volume, variety, and velocity, then the more Data Quality issues [32] (derived from the integration of data from different data sources, with different semantics and schemata), the more complex the reproducibility [36] is (since combining high amounts of heterogeneous data increases the number of operations to perform), and the more security issues might arise [51] (e.g., due to the combination of data with different security and privacy policies). The ultimate objective of the Big Data paradigm is to process this type of data to extract knowledge and value [26, 95, 93]. It is done through Big Data pipelines, which are processes composed of activities to extract value from data. This concept and the activities that make up it have been widely studied in the literature [22, 9, 64, 26].

Big data pipelines play a key role in the industrial context of today [56]. On the one hand, it is capable of providing support at the operational level by facilitating the automation of the processing and storage of transactions in business processes and is capable of processing large volumes of heterogeneous data. On the other hand, the Big Data pipelines are the cornerstone of the strategic and tactical levels of the organisations, since these enables the analysis of the transactional data in conjunction with third-party data.

In summary, Big Data pipelines are key to assisting companies in aspects such as decision-making in business processes.This thesis focusses on this type of Big Data pipelines and on the use of optimisation algorithms to facilitate decision-making processes.Therefore, we face the challenges derived from the distributed data with Big Data technologies to distribute the computation of optimisation problems so that optimal solutions can be efficiently obtained.

Within the optimisation paradigm, several types of algorithms can be found. These can be classified into metaheuristic algorithms [17] and exhaustive algorithms [33]. As we discuss later, there exist several proposals [14, 13, 1, 94] to support metaheuristic algorithms in distributed environments. However, we noted a lack of support for exhaustive optimisation algorithms. Furthermore, exhaustive optimisation algorithms do not scale well, since the

computation time significantly increases as the search space increases. This implies an important challenge in the current scenario, since the processing of large volumes of heterogeneous data is required, as well as the computation of more optimisation problems. These factors contribute to increasing search space, leading to scalability issues.

The next Section (Section 1.1.2) goes into more detail on the different areas that are subject to improvement in order to support this type of analysis in Big Data pipelines.

1.1.2 Motivation

This thesis focusses on the enhancement of those Big Data pipelines that are intended to solve large and complex optimisation problems to assist decision-making processes. We focus on exhaustive optimisation problems, which usually are NP-complete, increasing their complexity as the amount of data to be processed augments due to the increase in the size of the search space. Therefore, we seek to improve this type of analysis in Big Data environments. The optimisation problems can require different data type structures, depending on the optimisation model. For example, in this thesis, we tackle use cases whose optimisation models require data with complex structures (i.e., data with nested arrays or other structures) or very specific data formats. For this reason, we also pursue improving the Data Preparation stage by supporting the transformation of data with complex structures. The results of the optimisation problems are meaningfully influenced by the quality of the input data, requiring the use of Data Quality measurement and assessment techniques to ensure the quality of the results. However, Data Quality measurement and assessment must be performed in accordance with the context of data use. In this thesis, we also strive to enhance the way in which Data Quality is evaluated and modelled.

This thesis is developed within the IDEA Research Group[1]. This group has proven experience in the application of optimisation problems, business process modelling, process mining, and data governance. The need to use techniques based on the Big Data paradigm (and consequently, Big Data pipelines) is derived from research projects at the national and local levels, as well as from collaborative projects with private companies. Next, we present the list of research projects in which this thesis plays a key role:

- *Aether-US*[2] (PID2020-112540RB-C44). This project aims to improve the way data is managed by following a context-aware basis to support business processes.
- *ECLIPSE*[3] (RTI2018-094283-B-C33). This project focusses on improving Data Quality and Data Security in business processes.
- *METAMORFOSIS*[4] (US-1381375). This project aims to support digital transformation processes by improving data management, business processes, and security governance.

[1] IDEA Research Group: `https://www.idea.us.es/`
[2] Aether-US: `https://investigacion.us.es/sisius/sis_proyecto.php?idproy=33855`
[3] ECLIPSE: `https://investigacion.us.es/sisius/sis_proyecto.php?idproy=29615`
[4] METAMORFOSIS: `https://investigacion.us.es/sisius/sis_proyecto.php?idproy=33186`

- *COPERNICA*[5] (P20_01224). This project aims to improve business processes through data governance.
- *Clean Sky 2*[6] (P036-19/E08). This project is carried out in collaboration with Airbus. Its objective is to improve the diagnostic of business processes in the aircraft assembly process.

This thesis supports the data management and processing part of each of these projects. The ultimate objectives of data processing in these projects are (i) to facilitate decision-making and (ii) to diagnose business processes. optimisation techniques are typically employed for these purposes (e.g., assisting in the selection of an optimal delivery route that reduces the cost, finding an optimal schedule for an industrial activity, selecting optimal configurations, or discovering the deviation between process execution and process model). Since the use cases we handle require processing large volumes of heterogeneous data, we decided to employ Big Data pipelines to address them. However, we discovered that there exists a lack of support for optimisation techniques in Big Data environments.

Within the optimisation problem paradigm, the distribution of metaheuristic optimisation problems has been widely studied, and there exist several approaches [14, 13, 1, 94]. Metaheuristic optimisation algorithms [17] do not explore the entire search space. Instead, these employ a heuristic function to find quasi-optimal solutions in a limited time. Therefore, metaheuristic optimisation is not exact, and the results obtained from these types of algorithms can be considered *approximations* to optimal values. On the other hand, exhaustive optimisation algorithms analyse the whole search space and are able to return the optimal value or values. In this respect, we noticed a lack of proposals for the computation of large-scale exhaustive optimisation problems.

Constraint optimisation [73] is one of the most popular exhaustive optimisation paradigms. This paradigm entails techniques that allow us to establish mathematical models to optimise an objective function by taking into account a set of constraints. It has traditionally been of paramount importance in decision-making processes [35, 81, 25], since this paradigm allows us to model real-world problems through mathematical functions. However, this type of problem tends to be NP-complete [33]. In Big Data scenarios, this situation worsens, since large volumes of data could potentially increase search space. In order to face the distribution of Constraint Optimisation Problems (COPs), the distribution of the search space has been proposed as a solution, known as *Distributed Constraint Optimisation Problems* (D-COP) [42]. However, the algorithms that have been proposed are too complex, since these require sharing and synchronising the problem states and search spaces, not existing commercial solutions applied in Big Data context. In this thesis, instead of distributing the whole search space of these types of problems, we seek to distribute and compute individual non-complex problems. We focus on two particular cases.

- The distribution of small individual problems. We call this approach *Constraint Optimisation Problems with Distributed Data* (COPDD). The idea is that the user groups the

[5]COPERNICA: `https://investigacion.us.es/sisius/sis_proyecto.php?idproy=33720`
[6]Clean Sky 2: `https://investigacion.us.es/sisius/sis_proyecto.php?idproy=31890`

data in such a way that a Constraint Optimisation Problem is formed for each record of the distributed dataset. As an advantage, COPDDs can be solved with traditional constraint optimisation algorithms and solvers. However, there are two drawbacks. On the one hand, each record must have all the information necessary to build the COPs, which potentially implies the appearance of data with complex structures. On the other hand, some individual problems can generate bottlenecks, so techniques are required to reduce the domain of individual problems.

- The distribution of big complex optimisation problems.In these cases, the way the distribution is carried out is strongly dependent on the type of problem. As an example, we use the case of conformance checking [21], which is a process mining technique that is fundamental in the diagnosis of business processes, since it allows one to calculate the alignment between the execution of a business process and its model. Conformance checking techniques entail the use of optimisation algorithms in order to find an optimal value within a search space. In this thesis, we seek a methodology to solve conformance checking problems through the distribution of the event logs (i.e., the execution traces of a business process), the decomposition of business process models, and an algorithm to combine and distribute the execution traces along with the different parts of the business process.

For these optimisation techniques to be employed in Big Data pipelines, Data Preparation activities must be properly carried out. As we mentioned above, the use of COPDDs may imply the use of data models with complex structures (i.e., data with various levels of nested structures such as arrays). This is the case in the case studies that are faced in this thesis. In our research, we found that most proposals do not support this type of data in a straightforward way, since most of the proposals are specifically designed for table-orientated data models, requiring intermediate abstract operations to perform Data Transformations within the nested data structures. For this reason, we decided to contribute in this direction. Furthermore, referring to the Data Preparation process, the application of process mining techniques, such as conformance checking, requires that the event logs have a specific format (i.e., the XES format [44]). In this direction, we found the opportunity to contribute to the extraction of event log from semi-structured complex data, since we discovered that most approaches are focused on relational data. In conclusion, this thesis seeks to contribute to the Data Preparation activity by supporting the transformation of data with complex structures in both general and process mining contexts.

The research projects we mentioned also focus on Data Quality. This is of paramount importance in the context of data governance. Furthermore, the success of the Data Analysis stage depends on the quality of the data, especially when optimisation algorithms are implied. For example, inaccurate data or missing data values can cause errors during the analysis process, leading to failures in decision making or misdiagnosis of business processes [32]. Due to the importance of Data Quality, this thesis also seeks to contribute to this field. In particular, we intend to provide context-aware mechanisms to model Data Quality rules, as well as automate the evaluation of the usability of data. In this way, data records that are not usable

within a particular use case can be discarded, guaranteeing that the Data Analysis process is carried out only with those data records that meet established quality standards.

In summary, this thesis intends to improve Data Preparation, Data Quality, and Data Analysis techniques to support the processing of complex data, the extraction of event logs from complex semi-structured data, the modelling of Data Quality rules, the evaluation of the data usability, and the analysis of the data through optimisation techniques.

1.2 Background

This section outlines the main concepts that are closely related to this thesis. Section 1.2.1 defines the concept of Big Data pipeline and the activities that make up it. Next, Section 1.2.2 dives into the concept of Data Preparation, and describes several of the most relevant activities related to this stage. Afterwards, Sections 1.2.3 and 1.2.4 show the definition of Data Quality, and describe the decision rule modelling language on which we rely in our proposal to face up the Data Quality rule modelling. Finally, Section 1.2.5 delves into the Data Analysis activities and contextualises the optimisation problem paradigm.

1.2.1 Big Data Pipeline

The term *Big Data pipeline* is extensively studied in the literature [22, 9, 64, 26]. The consensual definition could be that a Big Data pipeline is *a process made up of a set of activities that, executed in a coordinated way, allow extracting value from the data*. In a preliminary work [88], we performed a thorough study of the literature on the different definitions and descriptions of Big Data pipelines, as well as the activities and stages that comprise them. Figure 1.1 shows the different stages that make up the Big Data pipelines, as well as the activity that could be carried out within each stage.

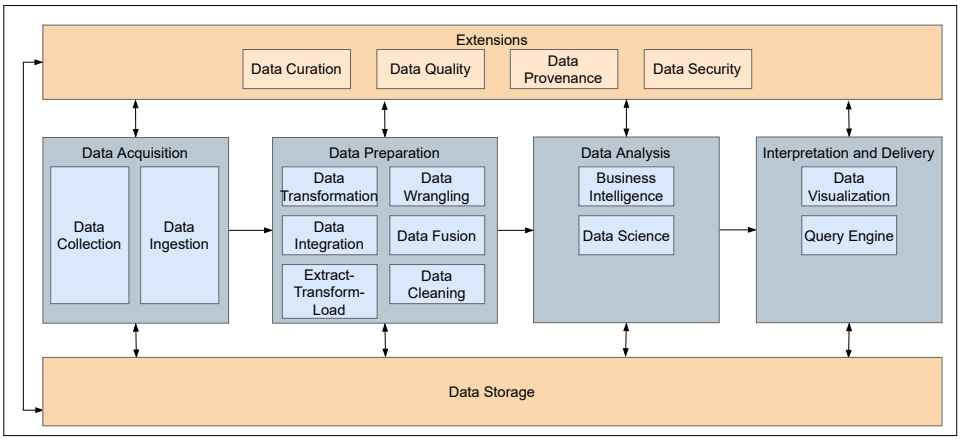


FIGURE 1.1: Big Data pipeline activities.

- **Data Acquisition**. This stage comprises the extraction and collection of data from different data sources. Activities within this stage involve data collection and ingestion [64,

22] (that is, data extraction from data sources and transportation to the preprocessing stages).

- **Data Preparation**. During the Data Preparation process, the raw data are prepared for the next processing activity in the pipeline. In short, Data Preparation consists of cleaning, formatting, transforming, and integrating data [22, 9].
- **Data Analysis**. Its objective is to extract value from the data. There are several activities and algorithms that can perform this task [26].
- **Interpretation and Delivery**. It is the final stage of a Big Data pipeline, and its objective is to report the value extracted from the data [22, 9, 26]. These results are consumed by end users or services. The way data are represented and delivered depends on the context of its use.

This Big Data pipeline framework also includes several additional activities:

- **Extension activities**. These activities are intended to provide support during the entire life cycle of the pipeline. These are: (i) Data Curation, which comprises the management of data during its life cycle [46] and entails monitoring the quality of the data and its enrichment; (ii) Data Quality, which involves monitoring the degree to which data satisfy requirements for its use [75]; (iii) Data Provenance, which is intended to keep track of the data during its life cycle [5], and; (iv) Data Security, which establishes the security requirements of the data and monitors their fulfilment during the life cycle.
- **Data Storage**. This is a transversal activity that provides persistence and access features when required by any activity.

There exist technologies that support modelling and executing Big Data pipelines. For example, Apache Spark [7], is a data processing engine that enables the programmatic implementation of data pipelines and workflows. This tool also implements functionalities to carry out certain activities of the Big Data pipelines. In this thesis, we focus mainly on Apache Spark and strive to enrich this tool by implementing the different methodologies that we devise.

1.2.2 Data Preparation

The Data Preparation stage is composed of activities that are intended to prepare the data for its analysis [9] by giving the data the appropriate format. In other words, it strives to transform raw data into a clean consumable dataset [48]. Next, we list typical activities that could be carried out in the Data Preparation stage.

- **Data Cleaning**. Data cleaning aims to identify and repair data values that fail to meet the Data Quality requirements of the use case [70, 45].
- **Data Integration**. This activity aims to provide unified access to data by combining data from different data sources [29, 48].

- **Data Transformation**. It is intended to facilitate data conversion and reformatting in order to adapt raw data to a specific data model [**Cong2009**, 48]. This process, which entails the transformation of the data schema and the modification and creation of data attributes, can be applied to structured, semi-structured, and unstructured data schemata.
- **Extract-Transform-Load (ETL)**. This is a paradigm that has traditionally been applied to data warehouses [30]. This concept refers to the integration, cleaning, and transformation of heterogeneous data with the objective of unifying the schema and format of data from several data sources.
- **Data Wrangling**. This concept has recently become popular [43]. This activity aims to provide Data Transformation mechanisms to non-expert users [41] so that they can transform structured, semi-structured, and unstructured data without high domain knowledge.

Specifically, we focus on the Data Transformation activity. Data Transformation has traditionally been used in data warehouses for Data Preparation through general purpose query languages or software to support ETL (Extract, Transform Loads) processes [23, 24]. However, ETL processes typically require specific and ad-hoc tools, as well as specialised programming skills. In [50], Joseph et al. remark on the differences between ETL and the new research concept of self-service Data Preparation [16], which is closely related to Data Wrangling. This activity has evolved into a new paradigm where the transformations are provided by non-expert users to perform end-to-end Data Preparation without programming knowledge. Currently, the efforts are directed towards facilitating and enriching the Data Transformations.

These lines of research are consistent with the need to speed up the design of Big Data pipelines. An easy implementation of Data Transformation stages is crucial in order to effectively test alternative solutions and compare their results or to adapt a pipeline to a new or evolving source of data [22]. In this direction, several tools have been produced. Trifacta Wrangler™[50] and other derived tools such as Google Cloud Dataprep are examples of solutions that enable Data Transformation. However, most of the solutions, including Trifacta, are specialised in table-orientated data models, requiring a number of intermediate abstract operations to modify data that are in deep levels of the data structure. In this context, it is of paramount importance to identify languages to facilitate the transformation of complex data structures [11].

1.2.3 Data Quality

Data Quality can be formally defined as a condition of data that can be measured and assessed through a set of characteristics or dimensions [15]. In the Big Data pipeline, Data Quality is typically used as part of the Data Preparation process [78] by providing insights that can be useful during Data Cleaning activities.

With respect to Data Quality measurement and assessment, we rely on the notion adopted by Juran [77] and I. Caballero [19], which states that the objective of the Data Quality assessment is to assess how usable data are in a specific context. On the other hand, Data Quality

measurement is focused on measuring how well data are built according to a set of requirements or rules. Regarding the Data Quality model, we rely on the one proposed in the international standard ISO 25012 [75]. This model states that the measurement of the Data Quality depends on one or several Data Quality dimensions (a.k.a. Data Quality characteristics). The characteristics to measure depend on the context in which the data are used. The *completeness* (i.e., the degree to which the data are complete), the *accuracy* (i.e., the degree to which the data are accurate) and the *consistency* (i.e., the degree to which data are consistent) are examples of Data Quality dimensions.

1.2.4 Decision Model and Notation (DMN)

DMN is a decision rule modelling standard [60] created by the Object Management Group (OMG). DMN offers a framework that enables the creation of decision rules visually, grouping them into decision tables (i.e., DMN tables). The decision rules are specified in different rows by following an if-then logic. In summary, the DMN tables are composed of three parts.

- **Inputs**. These represent the input attributes whose values will be evaluated;
- **Outputs**. These are intended to indicate the output value that would be produced if the decision rule is triggered.
- **Rows**. Each row specifies the set of conditions that the input values must meet to produce the output value indicated in that row.

Regarding the output, each DMN table can return one or multiple values depending on (i) the number of decision rules (rows) that are triggered and (ii) the hit policy specified in the DMN table. The hit policy enables one to define how to handle multiple matches. Some examples of hit policies are as follows.

- **First (F)**: In case multiple decision rules are triggered, return the output which corresponds to the first decision rule defined in the table.
- **Priority (P)**: In case multiple decision rules are triggered, return the one with highest priority.
- **Unique (U)**. Only one decision rule must be triggered.
- **Collect (C)**: In case several decision rules are triggered, return an aggregation of their corresponding outputs (see [60] for further hit policies and details).

DMN supports *expression languages* that facilitate the description of the conditions that data must meet. In our proposal, we rely on the FEEL[7] expression language. It is a versatile language that supports several data types (e.g., Integer, Decimal, Date, String, Boolean), and offers a set of built-in functions to write complex conditions. Its versatility also enables users to modify and write their own built-in functions. Finally, note that FEEL supports conditions that are always true ("-").

[7] `https://docs.camunda.org/manual/7.4/reference/dmn11/feel/`

1.2.5 Data Analysis and Optimisation Problems

The Data Analysis stage includes activities that use algorithms to extract value from the data [22]. These activities take properly structured and formatted data as input so that the algorithm can process them properly. Depending on the ultimate objective of the pipeline, the analysis can be [9, 67] descriptive (i.e., the objective is to identify phenomena and discover their causes), predictive (i.e., the objective is the prediction of future events), or prescriptive (i.e., the objective is to assist in decision making once an event has been detected through a descriptive or predictive analysis).

This dissertation focusses on the use of optimisation problems in the Data Analysis stage. These techniques are widely used in decision making scenarios [35, 81, 25] to help optimise resources, costs, or time. Optimisation techniques enable modelling real-world problems in the form of a set of variables and one or more objective functions that depend on that set of variables. These algorithms try to find a combination of values for the variables that maximise or minimise the value returned by the objective function.

With respect to optimisation algorithms, several techniques can be found in the literature. In summary, these can be classified into metaheuristic or exhaustive optimisation algorithms.

- **Metaheuristic optimisation algorithms**. These algorithms do not explore the entire search space [17]. Instead, these use heuristic functions use that guide the search towards local optimal values.

- **Exhaustive optimisation algorithms**. These algorithms explore the entire search space, and therefore are able to find global optimal values, guaranteeing the finding of optimal values.

The main drawback of exhaustive optimisation algorithms is the fact that these can become NP-complete [33] depending on the search space, negatively affecting to the scalability. However, metaheuristic optimisation algorithms can be more scalable in certain cases [94].

In this thesis, we focus on exhaustive optimisation problems, in general, and on Constraint Optimisation Problems (COPs), in particular. A Constraint Optimisation Problem [73] is a Constraint Satisfaction Problem (CSP) with an objective function that must be optimised according to the criteria defined by the user. Therefore, the optimal value reached by the objective function represents the *best solution* among all possible solutions.

Next, we include a formal definition of CSP, given by M. T. Gómez-López et al. [38]. A CSP represents a reasoning framework consisting of variables, domains and constraints $\langle V, D, C \rangle$, where V is a set of n variables $\{v_1, v_2, \ldots, v_n\}$ whose values are taken from finite domains $\{D_{v_1}, D_{v_2}, \ldots, D_{v_n}\}$ respectively, and C is a set of constraints on their values. The constraint c_k $(x_{k_1}, \ldots, x_{k_m})$ is a predicate defined on the Cartesian product $D_{k_1} \times \ldots \times D_{k_m}$. This predicate is true iff the assignment of the value of each variable x_{k_i} satisfies the constraint c_k.

Therefore, a COP is intended to explore the search space of the variables V, limited by their domain D and constraints C. This exploration strives to optimise an objective function which must be minimised or maximised (i.e., the objective could be finding the minimum or maximum value returned for this function). The objective function depends on the values

taken by the variables V. The values taken by each variable $v_i \in V$ must be within the domain of this variable (D_{v_i}), and must meet the predicates defined in each constraint c_k. Then, the best solution is the set of variables values that returns the optimal value.

1.3 Objectives

In this Section, the objectives of this thesis are presented. These are related to the enhancement of various activities in the Big Data pipelines to help companies process their data, emphasising those use cases that require using optimisation problems for decision making. Given the context in which this thesis is developed, the objectives traced are related to improving Data Preparation, Data Quality, and Data Analysis activities. Figure 1.2 summarises the three global objectives that we address.

- **OBJ 1**. This objective is related to the Data Preparation stage. In particular, it addresses the transformation of complex data and the extraction of event logs from complex semi-structured data.
- **OBJ 2**. This objective aims to improve the Data Quality activity by enhancing the way in which Data Quality rules are modelled, focusing on a context-aware basis. This objective also aims to promote the automation of data usability evaluation and the improvement of data usability repair.
- **OBJ 3**. The last objective is related to the Data Analysis stage. It aims to improve the computation of optimisation problems with distributed data.

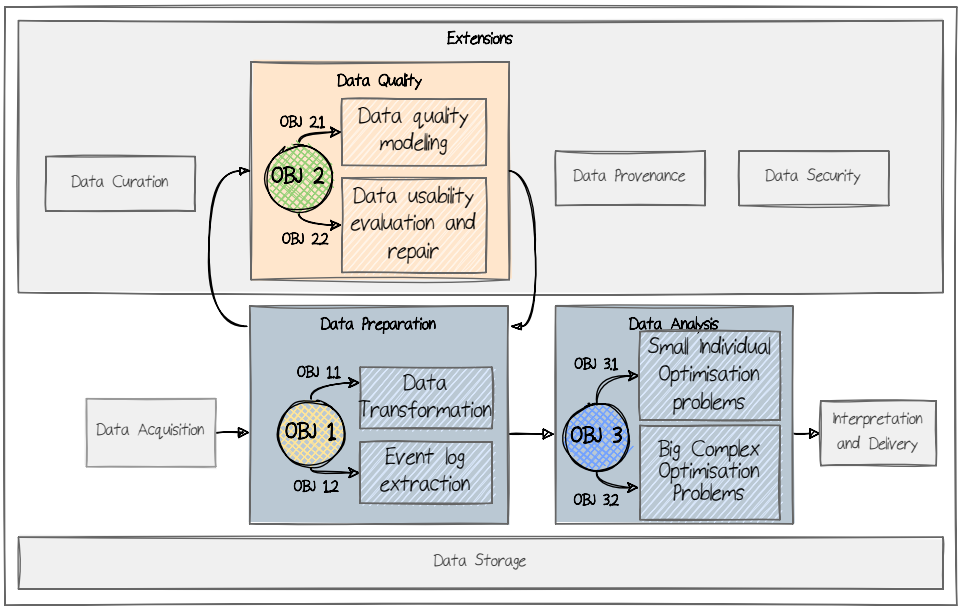


FIGURE 1.2: Summary of the objectives of this dissertation.

1.3.1 Objectives related to the Data Preparation activities (OBJ 1)

Data Preparation is a crucial stage in any Big Data pipeline since it gives the appropriate form to the data for later analysis. If Data Preparation activities are not carried out properly, probably the Data Analysis will fail.

The Data Preparation activities entail certain challenges in the current industrial context, since these activities require format, clean, integrate and transform large amounts of data from different data sources. Given this scenario, data structures tend to become more complex, requiring new techniques to transform this type of data [76] [49] [55]. In this scenario, Data Wrangling activity is ideal, as it helps transform, combine, and clean data in an exploratory way [34]. However, as mentioned in Section 1.2, most of the solutions are only specialised in table-orientated data models, requiring a number of intermediate abstract operations to deal with complex data.

Therefore, the first global objective of this thesis is to develop a framework and two languages to help transform data with complex structures. First, we intend to develop a general-purpose framework and a Domain-Specific Language (DSL) to fulfil the global objective. Second, and based on the general-purpose framework, we aim to develop a specific DSL to transform event logs with complex data structures into XES [44] event logs. These objectives are stated below.

- **OBJ 1**. Improve the Data Preparation process with complex data structures.
  - **OBJ 1.1**. Develop a DSL supported by a framework to transform complex data.
  - **OBJ 1.2**. Develop a DSL to support the extraction of event logs from semi-structured data.

1.3.2 Objectives related to the Data Quality activity (OBJ 2)

To succeed in the different activities of the Big Data pipeline, it is necessary that the data have appropriate quality levels [79]. Data Quality For example, in a decision-making Data Analysis activity where decisions are made at the record level based on its data, catastrophic decisions could be made if such data records contain errors. Data Quality This situation can be easily prevented if Data Quality is correctly evaluated.

According to [78], the quality of the data is strongly dependent on the nature of the data, its semantics, and the context in which it is used [10, 31]. Therefore, it is of paramount importance to model the Data Quality requirements together with the organisational context of the company. Additionally, in the current context, solutions to generate automatic recommendations on whether to use or discard the data are required [78, 74]. A solid context-aware solution to automatically measure and assess Data Quality would support many companies, as this process has typically been developed ad-hoc [79].

Another area of interest is improving the usability of the data [92]. This is usually done using data cleaning techniques [70, 45]. However, the application of these techniques must be in accordance with the results of the evaluation of the usability of the data and the organisational context.

Hence, the second global objective of this thesis is focused on the improvement of the evaluation of Data Quality in Big Data environments in two different ways. First, by developing a methodology to help model the Data Quality requirements in a context-aware basis, and second, by automating the generation of data usability recommendations and facilitating reparation of the data. Next, these objectives are listed.

- **OBJ 2**. Improve the evaluation of Data Quality.
  - **OBJ 2.1**. Develop a methodology to model Data Quality requirements in a context-aware basis.
  - **OBJ 2.2**. Develop a framework to facilitate the detection of the root causes of poor quality data, as well as data repair.

1.3.3 Objectives related to the Data Analysis activities (OBJ 3)

Data Analysis activities are essential to extract value from data. As discussed in Sections 1.1 and 1.2.5, there are a myriad of techniques to analyse the data and extract value from them. In this thesis, we decided to focus on optimisation techniques.

As mentioned in Section 1.1.2, we focus on two particular cases of optimisation problems. On the one hand, we intend to improve the distribution and computation of small individual optimisation problems in Big Data environments. Certain Data Analysis algorithms, such as some Constraint Optimisation Problems (COPs), tend to be NP-complete [33]. Therefore, the computing time increases significantly as the volume of data increases: This is an scalability issue. The use of Big Data to distribute COPs in a cluster provides the opportunity [62] to improve the efficiency of this type of algorithm when processing large amounts of data. As we also explained previously, some authors [42] face the scalability issue by distributing the constraints and the search space. However, there exists another approach that, to the best of our knowledge, has not been addressed to date: Instead of distributing the problem *entire*, divide the data into smaller groups of related data so that the size of each problem is more affordable. Therefore, the objective OBJ 3.1 seeks to provide a systematic mechanism to model, generate, and compute COPs in Big Data environments in an efficient way.

On the other hand, we also focus on big complex optimisation problems whose distribution is not trivial. In these scenarios, the way the distribution is performed strongly depends on the type of problem. Provided that this thesis is framed in the context of research projects that require the diagnosis of business processes, we propose the use of conformance checking techniques as a case study for this type of optimisation problems. These types of problem also tend to be NP-complete [21]. In short, conformance checking techniques are intended to relate models (business processes) and the observed behaviour (event logs from that business process), so that the deviations between the process log and the process model can be revealed. The alignment problem is the cornerstone of conformance checking. This can be seen as an optimisation problem that strives to find the end-to-end model run that most cosely resembles an observed event log trace. Alignment calculation is a highly complex problem, which requires exponential computation times [3], since it requires exploring the entire search space.

In a Big Data scenario, with massive event logs and complex business process models, the situation worsens due to the poor scalability of these algorithms.

Therefore, the objectives related to the Data Analysis activity are as follows.

- **OBJ 3**. Enhance the distribution and computation of optimisation problems in Big Data environments.
  - **OBJ 3.1**. Develop a framework to model, generate, and solve Constraint Optimisation Problems with Distributed Data.
  - **OBJ 3.2**. Develop a solution to compute big complex optimisation problems in Big Data environments applied to conformance checking techniques.

1.4 Research Methodology and Resources

1.4.1 Research Methodology

The *Design Science Research* (DSR) research methodology [65] has been followed during the development of this thesis. In summary, this methodology is intended to develop scientific-technical knowledge so that it can be applied in industrial scenarios by solving problems faced by professionals in those fields. The ultimate objective is to generate artifacts that solve an existing problem. Six activities [65] compose this methodology.

- **Activity 1: Problem identification**. First, a specific problem must be identified. It must be conceptualised and divided into multiple atomised subproblems. The relevance of the problem must justify the value that a solution could provide.
- **Activity 2: Define the objectives of the solutions**. Once the problem is identified, the objectives of the solution must be inferred.
- **Activity 3: Solution design and development**. This activities comprises the design and development of the artifact or artifacts that will be delivered as a solution to the problem. Artefacts can be methodologies, models or tools.
- **Activity 4: Demonstration**. Once the solution has been developed, experiments must be performed in realistic scenarios in order to assess the degree to which the solution is able to efficiently solve the problems detected.
- **Activity 5: Evaluation**. The solutions must be evaluated in realistic scenarios by solving specific use cases. With respect to this thesis, all published results have been evaluated in real-world use cases.
- **Activity 6: Communication of results**. The results obtained must be communicated. In this thesis, the different solutions were communicated through publications in relevant JCR journals, international conferences, industry forums, demonstrations of tools, and book chapters in relevant editorials.

Next, we describe the process we have carried out to complete the first activity (i.e., problem identification). This thesis arises from the need to use Big Data pipelines to process large volumes of heterogeneous data from multiple sources. Furthermore, in the Data Analysis stage, the use of optimisation algorithms was required to generate and solve optimisation problems from heterogeneous data. The first problem we identified was the lack of methodological tools to compute Constraint Optimisation Problems in Big Data environments. We denote this problem by **P-OP**. Since our optimisation models depended on data with complex structures, we proceeded to analyse traditional solutions to carry out this Data Preparation task. Data Quality On the other hand, we realised that the data of our case studies had flaws related to lack of completeness and accuracy. This affected the quality of the results that we obtained from the optimisation algorithms, since these are very sensitive to poor-quality data. After a review of the literature, we found the possibility of improving techniques for modelling and evaluating Data Quality in a context-aware way that would work in Big Data environments. We denote this problem as **P-DQ**.

Once these global problems have been identified, we have broken down each problem. Regarding the problem **P-DT**, related to the transformation of data with complex structures, one of the use cases that we handled for a private company required analysing its business processes. The data that contained the business process event logs were required to be transformed into a standard format (i.e., the XES event log format, which is widely accepted by the process mining community as the standard event log format). The data were semi-structured and contained nested structures. We did not find a solution that would allow us to transform these raw event logs into XES-formatted event logs in a straightforward way. Therefore, we identify two subproblems related to **P-DT**: (i) **P-DT-1**, which refers to the transformation of data with complex structures, and (ii) **P -DT-2**, which refers to the transformation of complex data in the standard XES format.

Regarding the problem **P-DQ**, related to Data Quality modelling and evaluation, we detected two areas of improvement, represented by the following subproblems: (i) the modelling of Data Quality rules by means of decision rules in a context-aware manner and (ii) the evaluation and repair of data usability.

Finally, in relation to the **P-OP** problem about the generation and computation of optimisation problems in Big Data environments, we have broken it down into two subproblems. The first, **P-OP-1**, deals with the generation and computation of large volumes of COPs. We faced this problem in a use case that required the creation and instantiation of an optimisation problem for each record in a dataset of more than $5,000,000$ records. The second subproblem, **P-OP-2**, is related to the computation of big complex optimisation problems. In particular, we faced the challenging scenario of solving conformance checking problems with complex process models and large volumes of event logs.

Hereinafter, this thesis is organised around the following three areas of improvement, related to the stages and activities of the Big Data pipelines that we seek to improve. Each area of improvement is called *scope*:

- *Data Preparation*. This scope groups those problems related to the Data Preparation stage within the Big Data pipeline (**P-DT**, **P-DT-1**, and **P-DT-2**).

- *Data Quality*. Within this scope, problems related to the Data Quality activity are included. (**P-DQ**, **P-DQ-1**, and **P-DQ-2**).
- *Data Analysis*. Finally, this scope groups the problems that are related to the Data Analysis activity (**P-OP**, **P-OP-1**, and **P-OP-2**).

1.4.2 Methodology Results

Data Preparation

Figure 1.3 describes the results of the different activities of the methodology that we have followed within the scope of Data Preparation.

- **Activity 2: Objectives traced**. In summary, the objectives related to the Data Preparation stage (**OBJ 1**, **OBJ 1.1**, and **OBJ 1.2**) are aligned with the group of problems **P-DT**.
- **Activity 3: Solutions developed**. Once the objectives were traced, we developed three different solutions to solve the group of problems **P-DT**, and therefore, fulfil the objectives related to them. The solutions are: (i) a general-purpose framework to facilitate complex Data Transformations (**S-DT-1**), and (ii) two Domain-Specific Languages that are based on this framework to support general-purpose complex Data Transformations and the extraction of event logs from semi-structured data (**S-DT-2** and **S-DT-3**).
- **Activities 4 and 5: Demonstrations and evaluations carried out**. Both **S-DT-2** and **S-DT-3** were evaluated through two different case studies. On the other hand, the computational efficiency of the framework (**S-DT-1**) was tested by benchmark.
- **Activity 6: Communications**. Finally, the results obtained were presented at two conferences. **S-DT-1** and **S-DT-2** were presented in *International Conference on Information Systems 2019*, with a GGS SCIE rank *Class 2*. **S-DT-3**, on the other hand, was presented at the Conference Industry Forum *Business Process Model 2019*. In relation to this presentation, in 2021, a book chapter was published in the book *BPM Cases Vol.2*.

The three aforementioned solutions are presented in Chapter 2 (Section 2.2). The results that have been produced with respect to the objectives **OBJ 1.1** and **OBJ 1.2** are discussed in Chapter 3.

Data Quality

Regarding the scope of Data Quality, Figure 1.4 describes the results of the different activities of the methodology.

- **Activity 2: Objectives traced**. The objectives related to the Data Quality activity (**OBJ 2**, **OBJ 2.1**, and **OBJ 2.2**) are aligned to the group of problems **P-DQ**. Hence, the solution to solve these problems must include a methodology to help model Data Quality rules in a context-aware basis and a framework to detect and repair poor quality data in Big Data pipelines.

- **Activity 3: Solutions developed**. The solutions that we developed to fulfil these objectives are: (i) a methodology to model Data Quality requirements in a context-aware basis (**S-DQ-1**); (ii) a tool to automate the evaluation of Data Quality models in Big Data environments (**S-DQ-2**), and; (iii) a framework to help repair and detect root causes of poor quality data.

- **Activities 4 and 5: Demonstrations and evaluations carried out**. The three solutions were evaluated in a scenario based on a catalogue of cloud server instances. In addition, **S-DQ-2** and **S-DQ-3** were tested using a benchmark.

- **Activity 6: Communications**. **S-DQ-1** and **S-DQ-2** were published in the *Decision Support Systems* journal, with a JCR rank within the first quartile (Q1). A preliminary version of **S-DQ-1** was published in a workshop (*DEC2H 2019*). Finally, **S-DQ-3** is yet to be published in the journal *Decision Support Systems*.

The aforementioned solutions are presented in Chapter 2 (Section 2.3). The results that have been produced with respect to objectives **OBJ 2.1** and **OBJ 2.2** are discussed in Chapter 3.

Data Analysis

Regarding the scope of Data Analysis, Figure 1.5 describes the results of the different activities of the methodology.

- **Activity 2: Objectives traced**. The objectives related to the use of optimisation problems in the Data Analysis stage (**OBJ 3**, **OBJ 3.1**, and **OBJ 3.2**) are aligned to the problem group **P-OP**. Therefore, the solutions that are intended to solve these problems must consist of a framework to model, generate, and compute Constraint Optimisation Problems, and compute big complex optimisation problems to solve use cases related to the conformance checking.

- **Activity 3: Solutions developed**. We developed two different solutions: (i) **S-OP-1**, a framework to compute Constraint Optimisation Problems with Distributed Data (COPDDs) in Big Data environments, supported by a tool (FABIOLA), and (ii) **S-OP-2**, a methodology supported by a tool (CC4Spark) to compute big complex conformance checking problems.

- **Activity 4 and 5: Demonstrations and evaluations carried out**. Each solution was tested using two different benchmarks in a distributed environment. Regarding the evaluation, both have been evaluated through two different case studies.

- **Activity 6: Communications**. **S-OP-1** was published in *Journal of Computational Science*, with a JCR rank within the second quartile (Q2). **S-OP-2** was published in the journal *Information Systems*, with a JCR rank within the third quartile (Q3). Furthermore, the tool that implements the solution **S-OP-2** was presented in the conference *Business Process Management 2021* within the *Tools & Resources* track.

Both solutions are presented in Chapter 2 (Section 2.4). The results that have been produced with respect to the objectives **OBJ 3.1** and **OBJ 3.2** are discussed in Chapter 3.

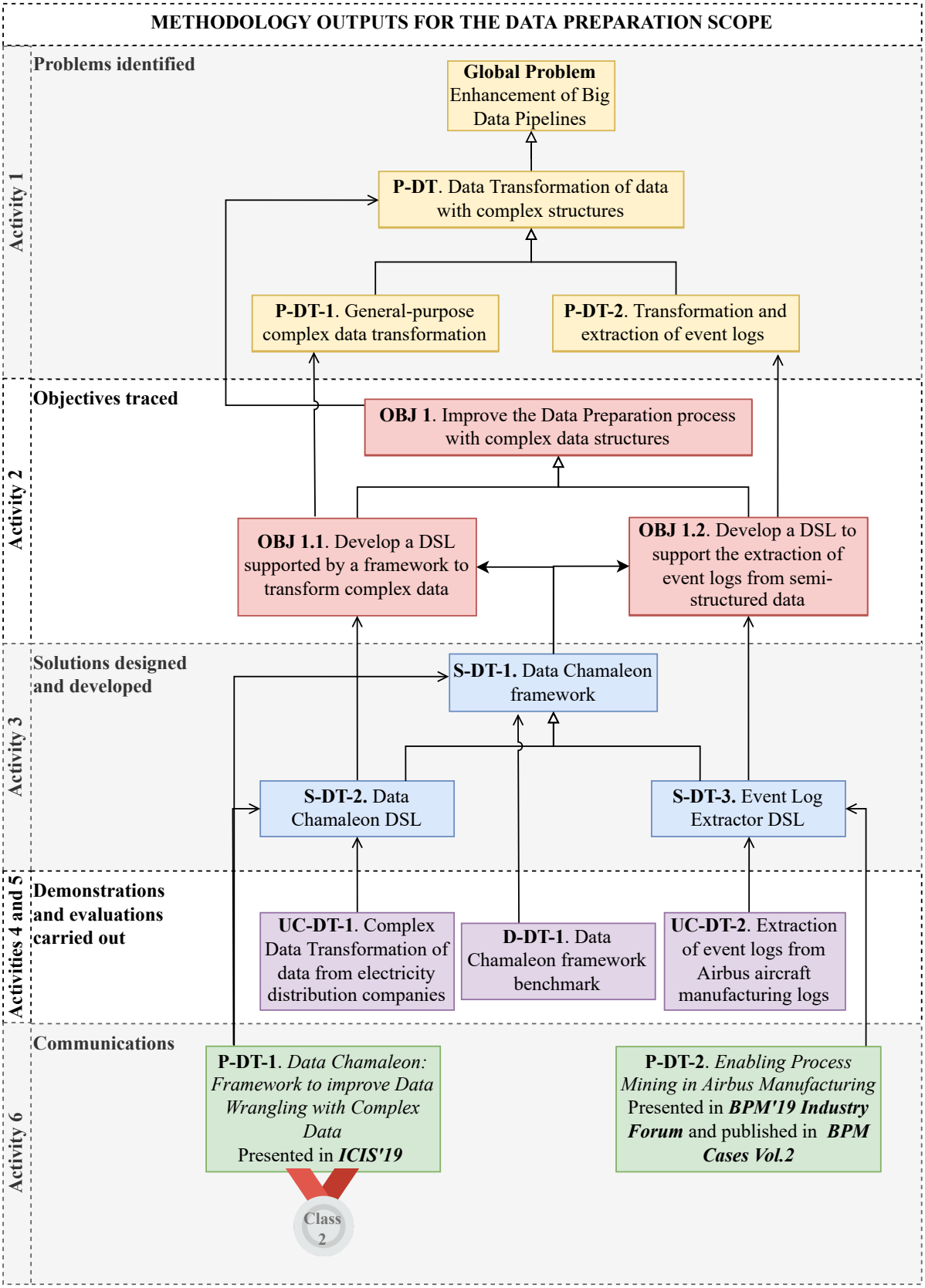


FIGURE 1.3: Methodology activities outputs related to the Data Preparation scope.

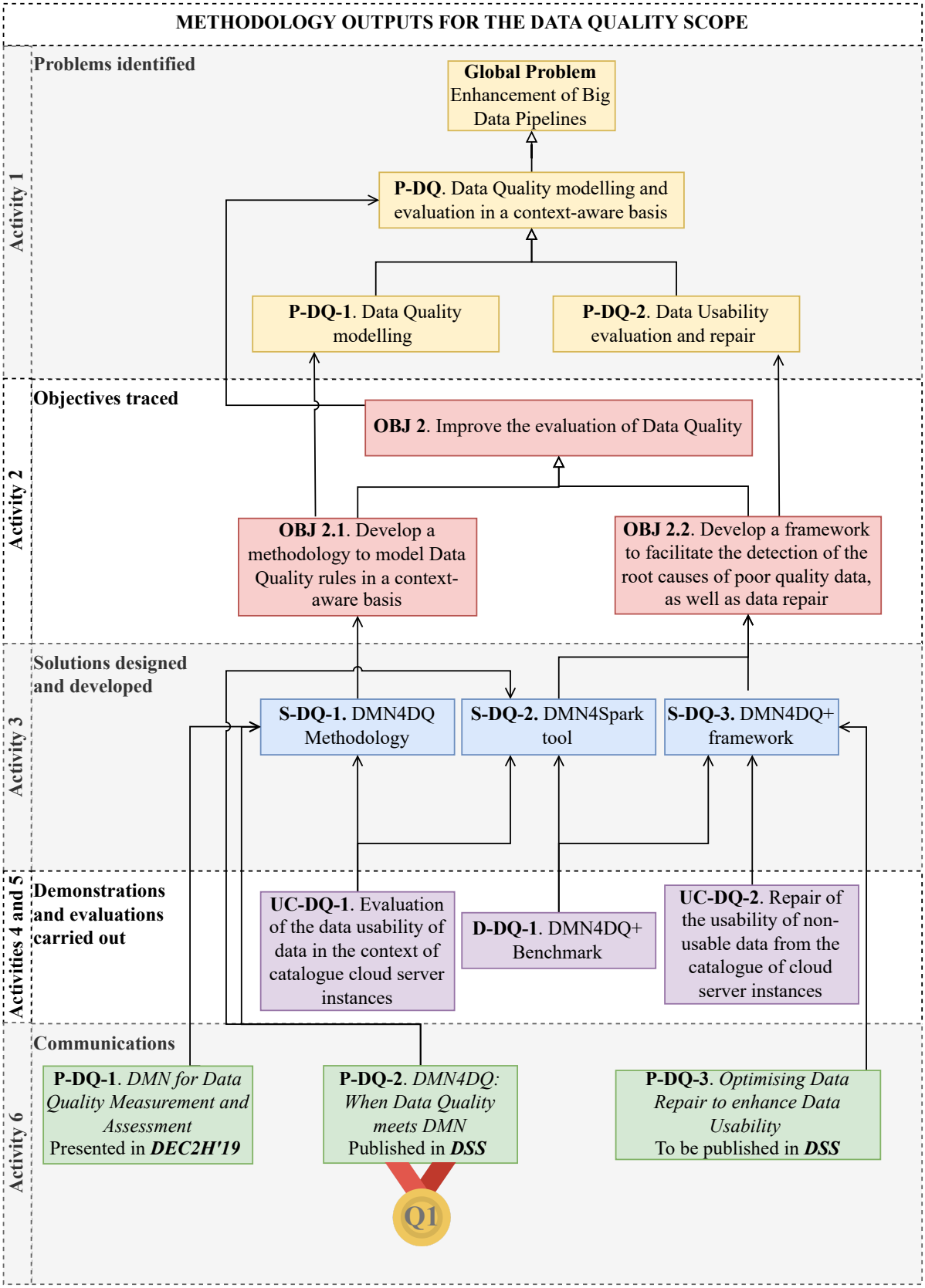


FIGURE 1.4: Methodology activities outputs related to the Data Quality scope.

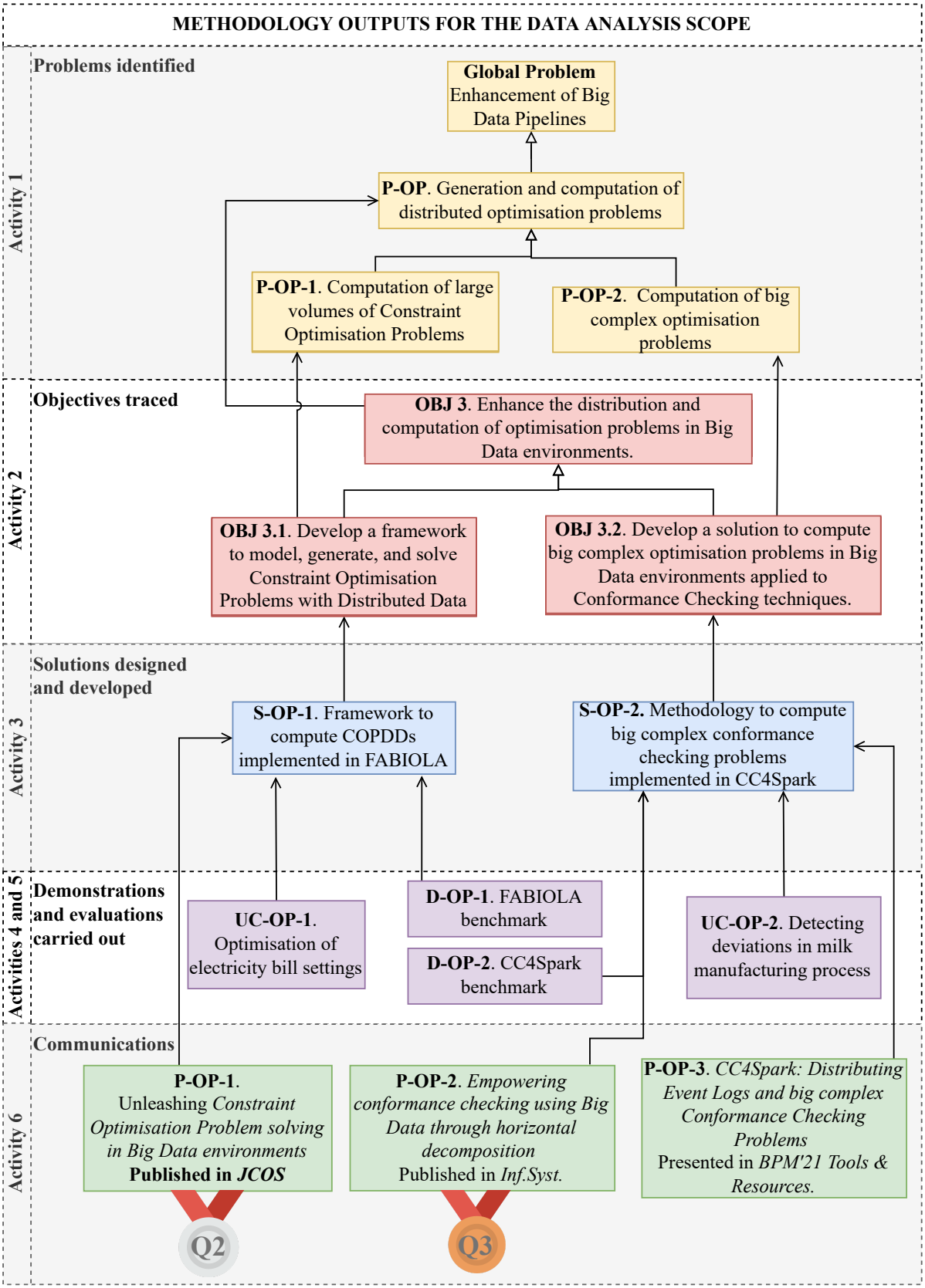


FIGURE 1.5: Methodology activities outputs related to the Data Analysis scope.

1.4.3 Resources

With regard to resources, the following have been used during the development of this research project:

- Scientific documentation. Research has been carried out through the study of the literature. The University of Seville has provided access to different scientific databases.
- Cloud servers and clusters. The IDEA Research Group has provided a private cloud. This cloud has been used to deploy big data processing and storage tools.
- Big Data processing tools, such as Apache Spark and Apache Kafka.
- Database management systems such as MongoDB and MySQL.
- Additional software to produce diagrams (such as draw.io, Microsoft Visio, Astah), documents (Latex through Overleaf, Microsoft Word), and presentations (Google Presentations, Microsoft PowerPoint).

All the software resources that have been employed are either open source or proprietary. The proprietary software has been employed through licences provided by the University of Seville.

1.5 Roadmap

The remainder of this thesis is structured as follows. Chapter 2 presents a summary of the results obtained for each objective defined in Section 1.3. Chapter 3 discusses the results and presents the publications that form and support this thesis. Finally, Chapter 4 draws conclusions from this thesis and discusses future work.

Chapter 2

Summary of the Results

This chapter presents the results of the research of this dissertation, introducing the different solutions that have been developed to fulfil the objectives defined. Section 2.1 gives an overall view of the results. Next, Sections 2.2, 2.3 and 2.4 describe and summarise the results related to the Data Preparation, Data Quality and Data Analysis stages, respectively. Finally, Section 2.5 presents two use cases in which the different solutions that have been proposed are employed.

2.1 Introduction to the results

As advanced in Chapter 1, the objectives of this thesis can be grouped into three large blocks: Data Preparation, Data Quality, and Data Analysis. Therefore, in this section, the results are presented following the same logic. First, in Section 2.2 the results related to the Data Preparation stage are presented: A Data Wrangling language to transform data with complex structures and a Domain-Specific Language to transform semi-structured data into XES event logs. Second, in Section 2.3 the results related to the Data Quality assessment activity are presented. There, we describe DMN4DQ, a context-aware methodology to assess the usability of the data. We also present a proposal to automate the Data Quality assessment, and a framework to help users repair their data. Finally, in Section 2.4, we present the results related to the Data Analysis activity, a solution to generate and solve Constraint Optimisation Problems in Big Data environments, and a solution to solve big complex conformance checking problems.

2.2 Data Preparation (OBJ 1)

This Section describes the solutions developed to accomplish objectives **OBJ 1.1** and **OBJ 1.2**.

- The solution **S-DT-1** (Data Chameleon framework) supports both **OBJ 1.1** and **OBJ 1.2**. The framework is presented in this Section.
- The solution **S-DT-2** (Data Chamaleon DSL) iis intended to satisfy **OBJ 1.1**. This is presented in Section 2.2.1.
- The Solution **S-DT-3** (Event Log Extractor DSL) fulfills **OBJ 1.2**. This is described in Section 2.2.2.

Our proposal to contribute to the Data Preparation with complex structures has two parts. On the one hand, we propose Data Chameleon [83], a general-purpose Data Transformation framework that includes a Domain-Specific Language (hereinafter, DSL) to transform data with complex schemata. Besides, we developed *Event Log Extractor* [86] (ELE), a DSL based on the Data Chameleon framework. ELE is designed for a specific domain: The transformation of business process event logs with complex data structures into a standard process mining format (XES [44]).

First, we must define what "complex data" means. Complex data are data structures that contain several levels of depth and consist of lists of data structures. According to our study[83], there is a lack of tools that facilitate Data Transformation with different depth levels. Traditional tools do not allow one to operate directly with these types of structure and require intermediate operations to bring the nested levels to the top level of the data schema. Data Chameleon intends to avoid these intermediate transformations, pretending that each operation applied to the source data has the desired final effect on the destination data.

Figure 2.1 shows an overview of the position of the Data Chameleon in a Big Data pipeline. The objective of it is to make the transformation of (complex) data from different data sources easier. In this example, different *transformation recipes* (i.e., *T1*, *T2* and *Tn*) are applied to unify the data structure of the data from different sources. This standardisation enables the application of Data Analysis algorithms.

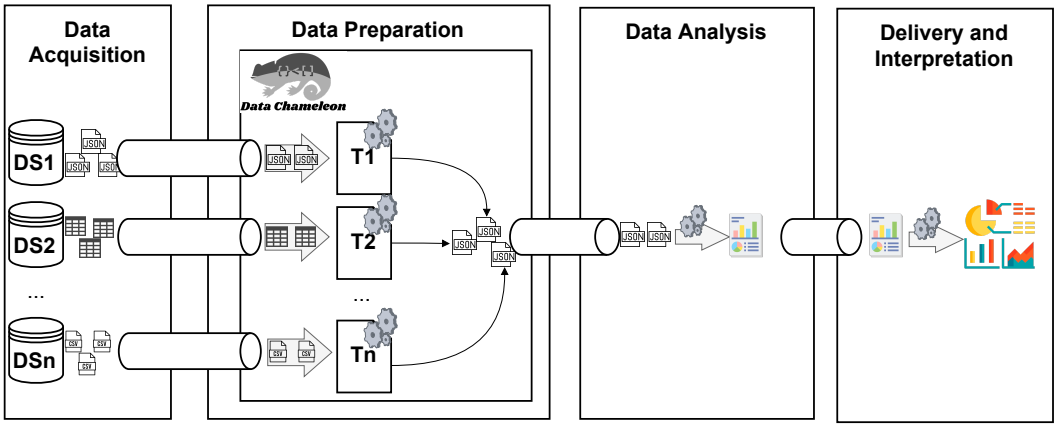


FIGURE 2.1: Data Chamaleon acts in the Data Prepration phase.

Data Chamaleon consists of a core framework that we have developed[1]. This can be extended to different DSLs. Figure 2.2 illustrates this differentiation. The Data Chameleon DSL is introduced in Section 2.2.1, while the ELE DSL is described in Section 2.2.2. With regard to the framework, it supports simple (i.e., String, Date, Boolean, Double, Float, Long and Integer) and complex data types, such as arrays (i.e., an indexed collection of typed attributes) and structures (i.e., a structure composed of a set of attributes with unique names). The implementation is supported by the Scala programming language [80], and is supported by a structure of classes following the composite design pattern [72], allowing complex objects to

[1]Data Chamaleon: `https://github.com/IDEA-Research-Group/Data-Chameleon`

be created from simpler ones. This is fundamental in our proposal, since it allows modelling the transformations of nested structures by means of composite classes.

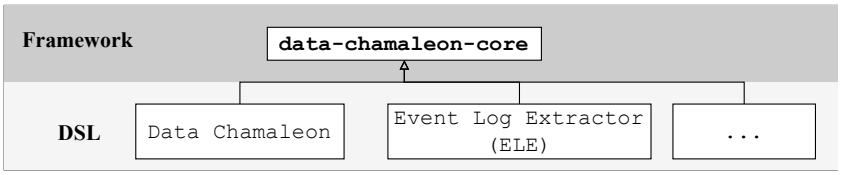


FIGURE 2.2: The Data Chamaleon framework can be extended by multiple Domain-Specific Languages (DSL).

2.2.1 Data Chameleon DSL: A general-purpose complex Data Transformation language

Next, we briefly describe the definition of the Data Chamaleon default DSL, extracted from our publication [83]. As mentioned above, this is a general-purpose DSL for performing all sorts of transformations. The definition is given through the Extended Backus-Naur form notation [71] in the Listing 2.1.

```
1  Syntax ::= Expression
2  Expression ::= Select|Index|Rename|CreateStruct|CreateArray|Iterate
3      |Transform
4  Select ::= 't' '"' ( StringLiteral | '[' Digit+ ']' )
5      ( '.' ( StringLiteral | '[' Digit+ ']' ) )* '"'
6  Rename ::= StringLiteral '<<' Expression
7  CreateStruct ::= 'struct' ' (' Expression ( ',' Expression )* ')'
8  CreateArray ::= 'array' '(' Expression ( ',' Expression )* ')'
9  Iterate ::= Expression 'iterate' Expression Transform
10 Transform ::= Expression '->' DTF
11 DTF ::= 'max'|'min'|'avg'|'sum'|'substract'|'toInt' |'toDouble'
12     |'toString'| 'toDate(' StringLiteral ')'
```

LISTING 2.1: Data Chamaleon DSL syntax definition.

This proposal was applied to two case studies [83]. For the sake of simplicity, in this summary, we show only one: The case study of Airbus, an aircraft manufacturer. The data contains information on the aircraft testing process. The dataset, which is given as a JSON file, contains information about the workstations (i.e., the stations where the aircrafts are tested). Each of them produces data on the tests that are carried out there, indicating the code of the workstation (*workstation*), the aircraft tested (*accode*), and a nested attribute indicating the list of incidents detected during the tests (*incidents*). Each incident is described with three attributes that indicate the date on which the incident began (*start_date*), the date on which the incident was resolved (*resolution_date*), and the type of incident (*incident_type*). Both date fields are represented as Strings with a specific format. Figure 2.3 depicts the source and target schemata. The transformations are as follows.

- **T1** and **T2**. Rename the attributes *accode* and *workstation* to *aircraft* and *ws*, respectively.
- **T3**. Create a new field *avg_resolution_time*, which represents the average resolution time.

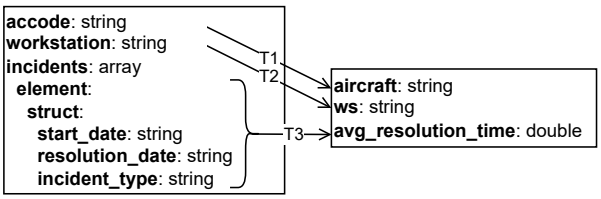


FIGURE 2.3: Left: Source schema. Right: Target schema.

The transformations are shown in Listing 2.2. Lines 1 and 2 perform transformations T1 and T2, while lines 3-6 perform transformation T3. In T1 and T2, the operator "<<" is employed to perform the rename operation. In T3, the array of *incidents* is iterated using the "*iterate*" operator. For each nested structure, the attributes *resolution_date* and *start_date* are transformed into date data types. This is done with the "*toDate*" transformation function, indicating the format of the date. Once transformed, both attributes are subtracted by means of the transformation function " *substract*". Finally, the average of the resulting array is calculated with the "*avg*" transformation function.

```
1 "aircraft" << t"accode"
2 "ws" << t"workstation"
3 "avg_resolution_time" << (t"incidents" iterate substract(
4     t"resolution_date" -> toDate("MM/dd/yyyy HH:mm:ss"),
5     t"start_date" -> toDate("MM/dd/yyyy HH:mm:ss")
6 )) -> avg
```

LISTING 2.2: BNF Notation

As we explained previously, CHAMELEON DSL aims to abstract the user from intermediate operations when performing complex transformations. In our study, we compare it to a leading Data Wrangling tool, Trifacta. Trifacta is a table-orientated tool that requires transformations to be performed at the top of the data structure. In Trifacta, this case study requires a total of seven operations, of which 4 (57%) are intermediate, that is, these are necessary in order to bring the nested structures to the top level of the schema (e.g., flattening, dropping, or renaming), increasing the difficulty of the transformations.

2.2.2 ELE DSL: A language to extract and transform event logs

In the current industrial context, business processes tend to employ a myriad of IoT sensors to monitor processes. Consequently, it favours the proliferation of event logs with different structures and formats. In this context, specialised techniques are required to extract and transform event logs into such a format that they can be employed during the use of process

mining algorithms. Within the Big Data pipelines, the extraction of event logs from data can be considered a Data Preparation activity, which strives to transform, format, and prepare the event logs for their consumption by process mining techniques such as conformance checking. In the literature, several authors agree on the importance of extracting event logs to proceed with the analysis of business processes [37, 12, 20, 57]. Some proposals focus on extracting event logs from relational data [57, 20], while others focus on XML, JSON, or other types of unstructured log files [37, 12]. However, we have not found any proposal focused on the transformation of complex structures into XES [44] event logs.

In this context, we propose ELE[2] (Event Log Extractor) [86], a DSL that enables the transformation of event logs into XES [44] event logs. It is the result of a collaboration with Airbus and aims to improve the extraction of event logs from IoT scenarios. Since it is based on the Chameleon framework, it is capable of processing event logs with complex nested structures. Figure 2.4 shows an example in which *extraction recipe* is applied to produce XES event logs. The XES [44] format is an IEEE standard (IEEE 1849-2016) that aims at standardising the way in which process logs are represented. In short, XES event logs are made up of the following fields.

- **Case ID**: It is the identifier of a specific instance or case of the process.
- **Activity**: It represents the identifier of a particular event and is associated with the case in which the event takes place.
- **Timestamp**: this is an attribute of the event that indicates the moment when the event occurred.

Log, traces, and events can have more attributes according to the standard, and ELE supports some of these, such as *Resource*, *TransactionType*, *Costs* or *Customer*.

Listing 2.3 shows the syntax definition of ELE using the Extended Backus-Naur form notation. It reuses elements from the Data Chamaleon DSL (*Expression* refers to the definition given in Listing 2.1 line 2). In this way, the trace ID (line 2) and the elements that compose the Event (lines 3-6) can be defined as a complex Data Transformation.

```
1 Syntax ::= 'extract (' Trace Event ')' From Save
2 Trace ::= 'define trace id(' Expression ')'
3 Event ::= 'define trace event(' ('activity = ' Expression) |
4     ('activity = ' Expression) | ('timestamp = ' Expression) |
5     ('resource = ' Expression) | ('transactionType = ' Expression) |
6     ('costs = ' Expression) | ('customer = ' Expression) ')'
7 From ::= 'from' String
8 Save ::= 'save' String
```

LISTING 2.3: ELE syntax definition.

Listing 2.4 shows an example of the use of this DSL. This is an extract of our work published in *Business Process Management Cases Vol. 2* [87], where we solve three different scenarios

[2]Event Log Extractor DSL: `https://github.com/IDEA-Research-Group/ELE`

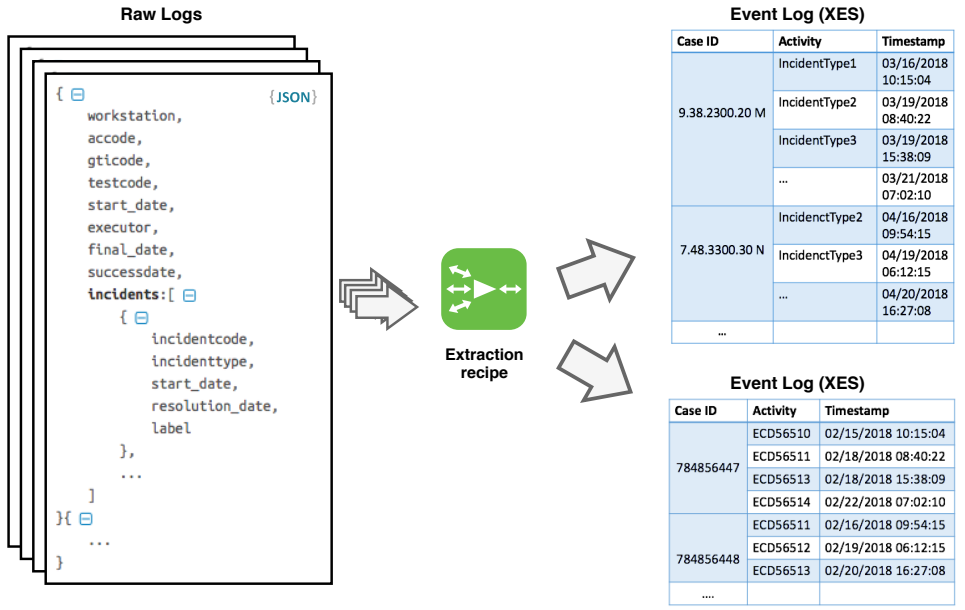


FIGURE 2.4: Event log transformation into the XES event log format.

to analyse the Airbus aircraft assembly process from different perspectives. This particular example is intended to produce an event log, in which the aircrafts are the traces and the workstations are the events. In this way, the global process describes how each aircraft passes each workstation.

- Line 2 indicates that the trace id is given by the attribute *accode*. Internally, this will result in a grouping operation using the Data Chameleon framework.
- Line 4 indicates that the activity (the event identifier) is given by the attribute *workstation*. In this way, the Data Chameleon framework will perform another grouping operation within each group (i.e., for each *accode*). Each workstation will represent a different event for each *accode*.
- Line 5 indicates the aggregation criteria for each event. By default, the first element of each group will be taken. In this example, an order operation is performed using the attribute *start_date* of each workstation, so the information relative to the moment when the plane arrived at the workstation for the first time will be taken.
- Line 6 indicates the attribute that will be used as a timestamp for each event. In this case, the attribute *start_date* has been selected.

```
1 extract(
2   define trace id(t"accode"),
3   define trace event(
4     activity = t"workstation",
5     criteria = orderBy(t"start_date" ->
6         toDate("MM/dd/yyyy HH:mm:ss")),
```

```
7       timestamp = t"start_date" -> toDate("MM/dd/yyyy HH:mm:ss")
8     )
9   ) from "datasets/aircraft_dataset_anonymized.json"
10      save "output/T1.xes"
```

LISTING 2.4: An example of transformation recipe.

After analysing event logs with a process discovery tool[3], experts in the business process can find deviations or abnormalities in the execution of processes.

2.3 Data Quality (OBJ 2)

This Section describes the solutions developed to accomplish objectives **OBJ 2.1** and **OBJ 2.2**.

- The solution **S-DQ-1** (DMN4DQ methodology) aims to achieve the objective **OBJ 2.1.** This is described in Section 2.3.1.
- The solution **S-DQ-2** (DMN4Spark tool) is part of **OBJ 2.2**. This is presented in Section 2.3.1.
- The Solution **S-DQ-3** (DMN4DQ+ framework) is intended to achieve **OBJ 2.2**. This is presented in Section 2.3.2.

2.3.1 The methodology: DMN4DQ

DMN4DQ methodology [89] is the solution we propose to model Data Quality rules in Big Data pipelines in a systematic and hierarchical way, and through an international decision-modelling standard to support the automation of data usability decisions. The rules enable the evaluation of the data set record by record. The methodology is supported by a tool that automates the Data Quality evaluation process. In summary, DMN4DQ strives to support the decision-making regarding the usability of each data record.

DMN4DQ takes into account the context of use of the data, which is influenced by the context in which the data are to be employed, the context in which the data were generated, and the context of the organisation. The data validation, measurement, and assessment processes vary depending on the context and the Data Quality dimensions to be measured. We propose that the context be modelled through a set of hierarchical business rules, where data stewards must describe the validation, measurement, and assessment rules that the different attributes of the data set must meet. In this methodology, it is key to differentiate between validation, measurement, and assessment. Validation is the degree to which the data values meet certain conditions. Measurement allows us to measure, for each Data Quality dimension, “how well data are built", and it is based on the degree of fulfilment of a set of data validation rules. Assessment, on the other hand, refers to “how usable the data are” in terms of a given context, taking into account the measurement values of each Data Quality dimension. To model and

[3]Disco by Fluxicon: https://fluxicon.com/disco/

implement Data Quality rules, DMN4DQ proposes the use of Decision Model and Notation (DMN) [60], which was presented in Section 1.2.4.

Figure 2.5 depicts the methodology, which is composed of three phases. Next, we briefly summarise the methodology, which is fully defined in [89].

The first phase entails the definition of business rules for Data Quality. The steps to follow require one to define the context in which data are used and define the following business rules following a bottom-up approach (i.e., starting more focused on the data values and increasing the level of abstraction until producing a recommendation about the data usability).

- **Business Rules for Data Values (BR.DV)**: These enable the validation of data values by evaluating the degree to which data values meet certain requirements.
- **Business Rules for Data Quality Measurement (BR.DQM)**: First, the Data Quality dimensions must be selected in accordance with the context of use of the data. Different measurement values must also be selected, which can be defined by Likert scales, each with a specific semantic. Then, BR.DQM produces a measurement value for each Data Quality dimension by *measuring* the degree to which each data attribute satisfies the BR.DVs.
- **Business Rules for Data Quality Assessment (BR.DQA)**: These business rules are intended to produce an assessment value for each data record, depending on the measurement obtained for each Data Quality dimension. The possible assessment outputs must be previously defined, each with a specific semantics, for example: “usable", if the data record can be used; “use with caution", if the data can be used but only in certain non-critical scenarios, or “non-usable", in case the data are not usable because the measurement values do not meet the Data Quality standards for the use case.
- **Business Rules for Data Usability (BR.DUD)**: These are intended to produce a recommendation on the usability of each record depending on the assessment value obtained for that record.

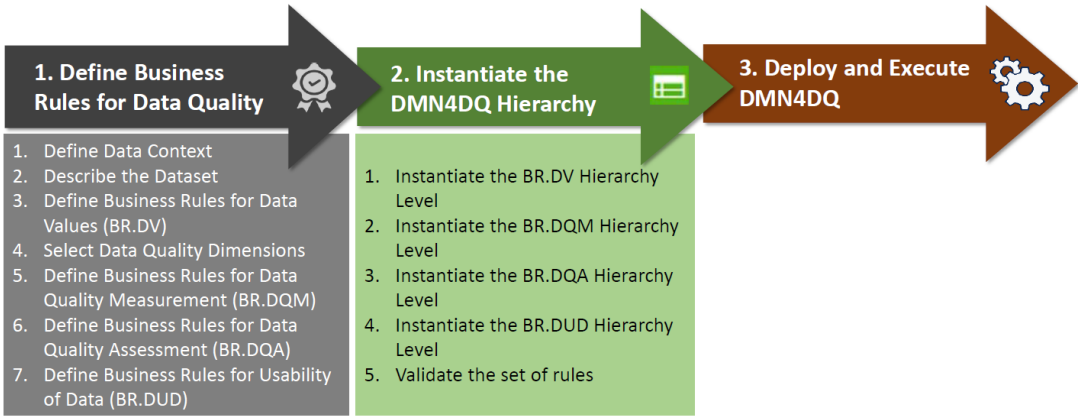


FIGURE 2.5: DMN4DQ methodology outline.

The second phase of DMN4DQ is to instantiate the DMN4DQ DMN hierarchy, which is shown in 2.6. Each level in this hierarchy implements the Business Rules defined during the first phase. These are:

- **BR.DV**, the DMN tables that describe each Business Rule for Data Value.
- **BR.DQM**, the DMN tables that describe each dimension of Data Quality.
- **BR.DQA**, the DMN tables that implement the Business Rules for Data Quality Assessment.
- **BR.DUD**, a DMN table that implements the Business Rules for Data Usability.

This hierarchy enables the evaluation of the usability of the data. The data values are the input of the hierarchical level BR.DV. Then, the output of the evaluation is propagated through the hierarchy until a recommendation about the usability of the data record is produced at the last the hierarchical level (BR.DUD).

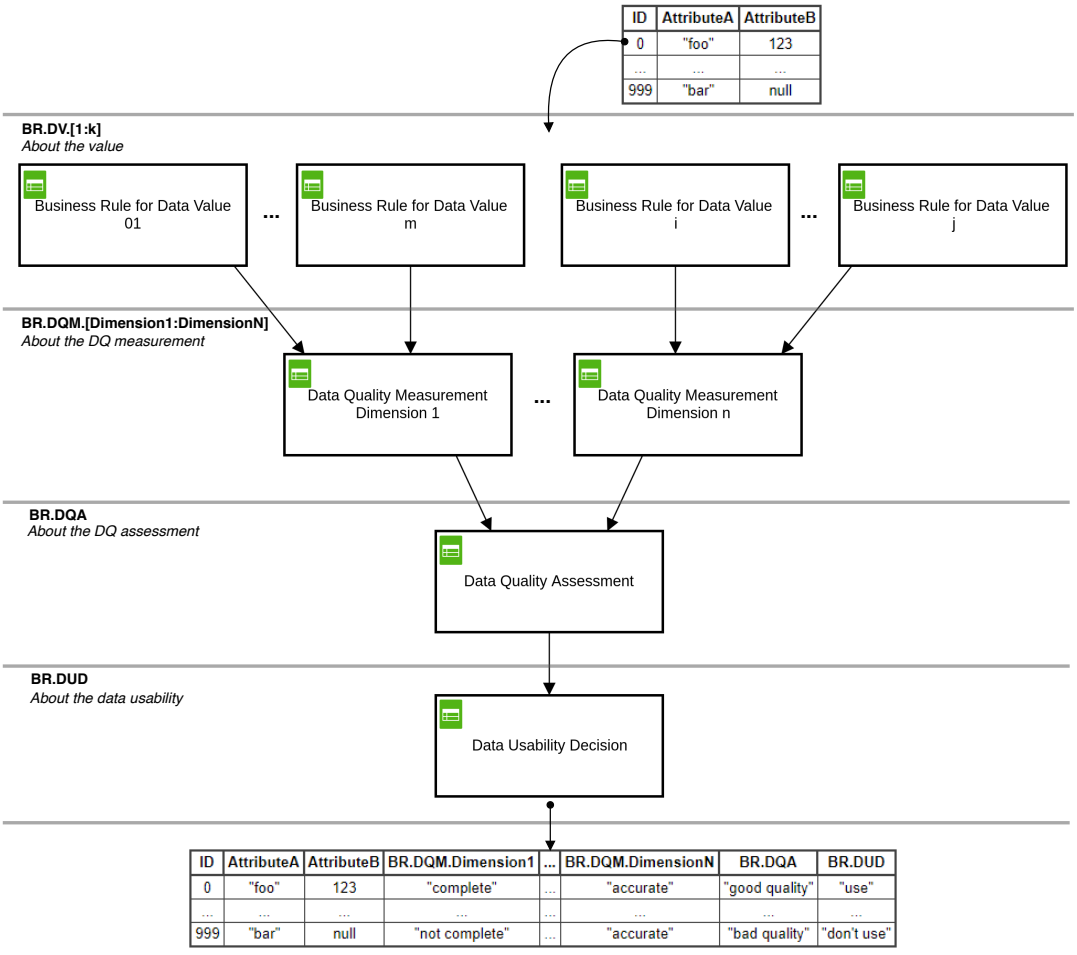


FIGURE 2.6: DMN4DQ DMN hierarchy.

Finally, the third phase of DMN4DQ comprises the deployment and execution of the decision model. This is supported by a tool that we have developed: DMN4Spark[4], a plugin for

[4]DMN4Spark: `https://github.com/IDEA-Research-Group/dmn4spark`

Apache Spark[7] that supports the execution of the Camunda Decision Engine on datasets in Big Data environments.

To illustrate this proposal, we present a small excerpt from the case study presented in our publication *DMN4DQ: When Data Quality meets DMN* [89]. The original dataset represents multiple cloud server instances provided by third parties, mainly from Amazon Web Services[5]. It contains information on the features of the servers, such as memory, CPU speed, type of storage, or network connection. In this small example, we focus on two attributes: *Memory* and *ClockSpeed*. The former represents the amount of memory available, while the latter indicates the speed of the CPU. The data are employed to offer a catalogue of server instances and are required to be at least complete and fairly accurate. Figure 2.7 depicts a small portion of the Data Quality model presented in the case study in [89]. There are three Business Rules for Data Values (BR.DV). The first refers to the completeness dimension and asserts that *ClockSpeed* and *Memory* are not *null*. The second and third BR.DVs are related to the accuracy dimension, and both assert that *ClockSpeed* and *Memory* follow a specific pattern. Following the next hierarchy level (BR.DQM), there are two dimensions of Data Quality. Completeness measurement can return two different values: *complete*, if BR.DV.01 is satisfied, or *not complete* otherwise. On the other hand, accuracy can return a numeric value: 100 if BR.DV.02 and BR.DV.03 are met; 50 if BR.DV.02 or BR.DV.03 are met, or 0 otherwise. The next hierarchy level (BR.DQA) produces an assessment value. Data Quality is assessed as *suitable* if completeness is *complete* and accuracy is 100. On the other hand, it will be assessed as *sufficient* if the completeness is different from *not complete*, and the accuracy is at least 50. Otherwise, the assessment would be *bad*. Finally, the Business Rule of Data Usability (BR.DUD) recommends using the data record if the assessment is *suitable* or *sufficient* and does not use the data record otherwise.

The Listing 2.5 shows an example of the deployment and execution of DMN4DQ. The dataset is a CSV file, and the Data Quality model is a DMN file. Both are located on a Hadoop HDFS server. First, the DMN4Spark library is imported (line 1). Second, the data set is loaded with Apache Spark (line 2). Finally, the Data Quality model is loaded and executed (line 3).

```
1 import es.us.idea.dmn4spark.spark.dsl.implicits._
2 val df = spark.read.csv(hdfs://hdfs-server.com/path/to/dataset.csv)
3 df.dmn.hdfs("hdfs://hdfs-server/path/to/dmn-file.dmn").load.execute()
```

LISTING 2.5: Executing the Data Quality model on a dataset.

The execution results in the evaluation of the usability of each record within the dataset. It allows users to obtain an overall view of the usability of their dataset. For example, in the case study presented in [89], we concluded that 48% of the records were not usable mainly due to accuracy issues. We could also identify the BR.DVs that most failed to be fulfilled, and hence, we could identify the type of Data Quality errors that each attribute presented.

[5]The data that has been used in this case study: `https://www.kaggle.com/akashsarda/aws-ec2-pricing-data/version/1`

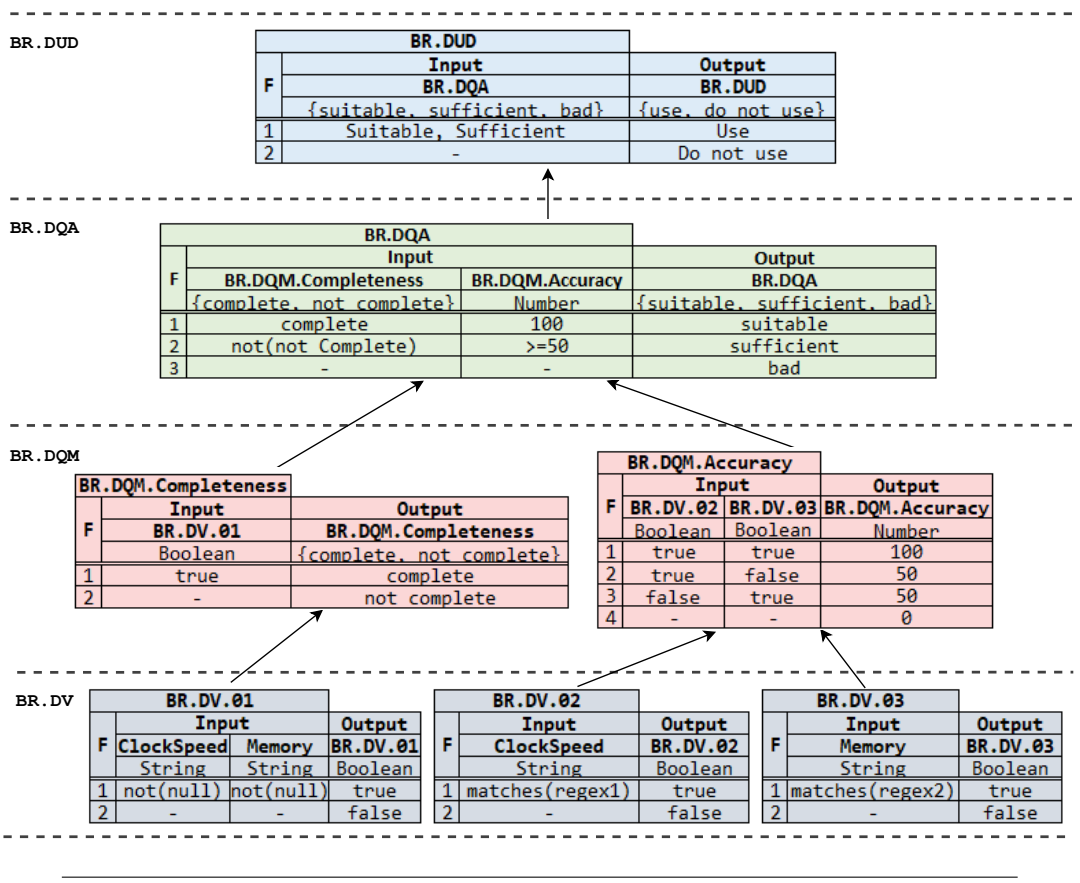


FIGURE 2.7: Example of an instance of DMN4DQ.

2.3.2 The extension: DMN4DQ+

With DMN4DQ, we proposed a methodology to support decision-making about the usability of each data record. This was implemented in a tool (DMN4Spark). In this Section, DMN4DQ is presented, an extension that aims to facilitate the diagnosis of the Data Quality of a dataset at a global level, offering a detailed view of quality to facilitate data repair. It also supports decision-making regarding data usability repair. The proposal has not yet been published.

A Usability Profile (up) is the characterisation of the usability of a record or a set of tuples in a dataset. It is described through a data structure $\langle decision, assessment, measurements, observations \rangle$, where:

- $decision \in D.BR.DUD.ouput$
- $assessment \in D.BR.DQA.output$
- $measurements$ = {BR.DQM.D1: m_1, BR.DQM.D2: m_2, . . ., BR.DQM.Dn: m_n}, being $BR.DQM.Di$ the name of the table for the quality dimension, so that $m_i \in$ D.BR.DQM.Di.output
- $observations$ = {BR.DV.V1: v_1, BR.DV.V2: v_2, . . . , BR.DV.Vm: v_m}, being BR.DV.Vi the name of a table defined in the $BR.DV$, so that $v_i \in$ D.BR.DV.Vi.ouput.

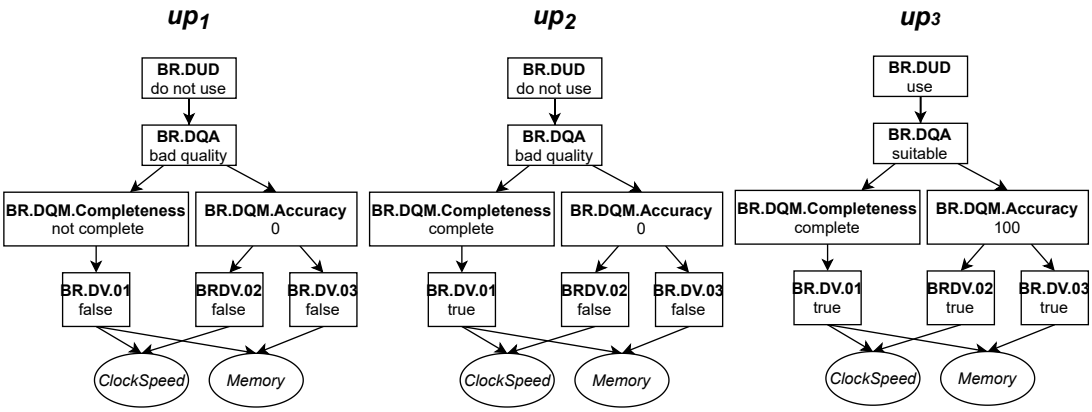
This data structure enables the representation of the usability of the data in a graph. This graph is generated from the results of the evaluation of the usability of the data. Therefore, for

each data record, this graph is generated as follows. First, the nodes are built hierarchically as follows.

- Decision. This is the root node, and its value represents the output of the evaluation of the DMN table BR.DUD.
- Assessment. This node is the child of the Decision node, and its value represents the output of the DMN table BR.DQA.
- Measurement. These nodes are children of the Assessment node. There is a node for each Data Quality dimension, and their value are the output of the BR.DQM DMN tables.
- Observations. These nodes are the children of the Measurement nodes. There is a node for each BR.DV table, and its values correspond to the output of these tables.

Regarding the edges, these have the following semantics: There is an edge between a node n and its children nodes if the values represented by the children nodes produce the output represented by n.

Following the example in Figure 2.7, we have calculated the usability profiles using that model and the dataset we used in our work [89]. This dataset contains $1,048,571$ records and produced 29 different usability profiles, of which 22 had the usability decision “do not use". In Figure 2.8, we show three of these usability profiles, of which up_1 and up_2 are classified as “do not use", and up_3 is classified as “use". There are $97,120$ records with the usability profile up_1, $53,247$ with up_2, and $221,147$ records with the usability profile up_3. This representation facilitates the diagnosis of the root causes of poor-quality data.



(A) Usability profile up_1. (B) Usability profile up_2. (C) Usability profile up_3.

FIGURE 2.8: Usability profiles in the guiding example.

DMN4DQ+ also proposes a methodology to help users find an optimal set of corrective actions to repair their data. This methodology requires users to relate each BR.DV to a specific Data Quality problem (i.e., a problem derived from non-compliance with business rules). Data Quality problems have been widely studied in the literature [92, 59, 6]. P. Woodall et al. [92] proposed a set of techniques to solve each type of Data Quality problem. In our methodology,

we studied the classifications of these authors and formalised a list of Data Quality problems that can be detected with DMN4DQ, and a set of *corrective actions* associated with each Data Quality problem. Regarding Data Quality problems, since DMN4DQ performs the evaluation record-by-record, the Data Quality problems that can be detected are those that comprise one or multiple attributes in a single tuple (e.g., syntax violation or violation of functional dependency). DMN4DQ+ allows the user to define a *catalogue of corrective actions*. This catalogue includes, for each corrective action:

- The **Data Quality problem** to solve (e.g., violation of functional dependency).
- The **corrective action** to apply (e.g., data rules).
- The **attributes** that would be affected by such action.
- The **operation** to perform (e.g., in this case, the specific data rules to apply).
- The **consequences** of the application of the corrective action (i.e., the user must specify the new outputs of the BR.DVs affected by the corrective action).
- The **requirements and conditions**, which include the output values required for each BR.DV before applying the corrective action.
- The **cost** of applying the corrective action. This cost must be set by the user using a Likert scale, which weighs the cost in human and computational terms.

Once the catalogue of corrective actions has been defined, the user can specify one usability profile to repair (e.g., a usability profile whose usability decision is “do not use"). We have developed a Constraint Optimisation Problem to select the set of corrective actions that modify the usability profile so that the usability decision is “use" with the optimal cost. The Constraint Optimisation Problem takes the following inputs: (i) the Data Quality model (i.e., the DMN model) in order to transform the DMN rules into constraints; (ii) the catalogue of corrective actions in order to build the constraints and variables, and (iii) the usability profile to repair. The variables are intended to select the corrective actions to perform, while the constraints allow one to bound the eligible actions by checking their feasibility in accordance to the consequences and conditions of the corrective action and the Data Quality model.

2.4 Data Analysis (OBJ 3)

This Section describes the solutions developed to accomplish objectives **OBJ 3.1** and **OBJ 3.2**.

- The solution **S-OP-1** (Framework to compute COPDDs implemented in FABIOLA) supports the objective **OBJ 3.1**, and is presented in Section 2.4.1.
- The solution **S-OP-2** (Methodology to compute big complex conformance checking problems implemented in CC4Spark) is intended to satisfy the objective **OBJ 3.2**. This is presented in Section 2.4.2.

2.4.1 Constraint Optimisation Problem solving in Big Data environments

Our proposal is motivated by the importance of Constraint Optimisation Problems (hereinafter COP) within the decision-making paradigm. Formally, a COP is a reasoning framework that consists of input data, variables, constraints, and an objective function. The COP algorithms are intended to optimise the objective function either by maximising or minimising it by producing permutations of values that the variables take. These values are bounded within a specific domain and must meet the defined constraints. Since COP algorithms are ultimately implemented through search algorithms, certain COPs can become NP-complete [33]. Furthermore, the complexity of the COPs increases as the amount of data increases. This complexity is motivated by an increase in the amount of input, variables, and constraints. In this distributed context, Distributed Constraint Optimisation Problems (hereinafter DCOPs) emerged [42]. These were intended to solve COPs where constraints, variables, and data were not in a single system. The algorithms must then be able to share and synchronise the problem states and search spaces. However, DCOPs are not focused on Big Data environments and do not provide mechanisms to enhance the distribution of COPs.

Therefore, our proposal focusses on improving the computation of individual COPs in distributed environments and optimisation of the execution of queries by distributing the COPs in chunks of data. Next, we discuss the concept of Constraint Optimisation Problem with Distributed Data (hereinafter COPDD), which summarises our contribution in [90]. A COPDD is applied to a dataset DS, and produces a dataset DS^{output} that includes the results of the COPs for each chunk of data. Therefore, a COPDD is composed of the following elements: $\langle$COP, DS, DM, $DMAP_{DS \rightarrow DM}$,$DMAP_{DM \rightarrow COP}\rangle$.

- COP. This represents the model of the Constraint Optimisation Problem to be computed for each piece of data.
- DS. The dataset that is employed to build the COPs.
- DM. This serves as the data model which specifies the relationships between the attributes of the dataset (DS) and the COP model (COP).
- $DMAP_{DS \rightarrow DM}$. This element alludes to the *data mapping* (i.e., the attribute alignment) between the dataset attributes (DS) and the attributes of the data model (DM).
- $DMAP_{DM \rightarrow COP}$. This refers to the *data mapping* between the attributes of the data model and the COP model.

Figure 2.9 summarises the phases that we propose for the generation and resolution of COPDDs.

- **Data Preparation**. At this stage, the datasets are integrated and transformed so that they can be mapped to the data model (DM).
- **COP description**. It contains the definition of the COP (i.e., the inputs, variables, constraints, and objective function).

- **Data Mapping**. This task is intended to map or *align* the attributes of the dataset, the data model, and the COP.
- **Data Querying**. This enables querying on the input and output data obtained from the computation of the COPs.

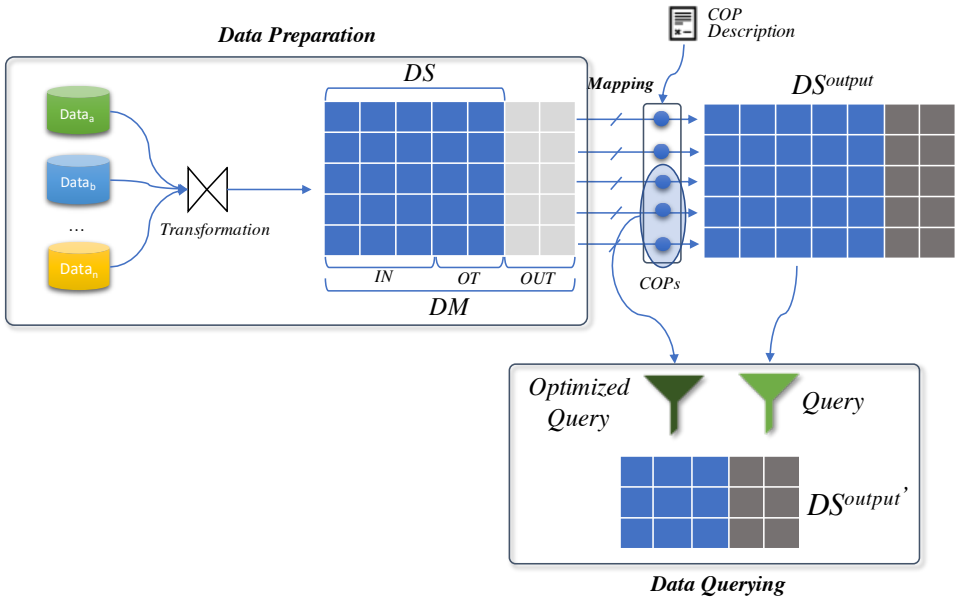


FIGURE 2.9: Overview of the approach to compute COPDDs.

The data model is divided into three groups of attributes: (i) *IN*, which represent the attributes that will be linked to the inputs of the COP; (ii) *OUT*, which are the attributes that represent the output values after the execution of the COPs, and (ii) *OT*, which includes those attributes that provide additional information that are not related to the COP but can be useful for querying tasks.

To illustrate the proposal, a real-world scenario is presented. This is part of the case study that we solved in the article in which we presented this proposal [90]. In this scenario, there are three Spanish electricity distribution companies that need to generate recommendations about the tariff configuration of their customers. For this purpose, they use COPs to obtain a hired power configuration that minimises the amount of the electricity bill, using the consumption data of each client as input. This scenario can be solved using COPDDs. An example is shown in Figure 2.10, representing one of the three datasets (DS), the data model (DM), and an excerpt of the COP model. Within the curated datasets, each record provides information about the contract and the consumption of the customer during different billing periods. Consumption is modelled as an array of structured data. Within each structure, the initial and final dates of the billing period are specified, along with the power consumed in three different periods of the day. Regarding the COP model, it requires as input the type of customer tariff (T) and a matrix C in which each row represents the different billing periods, and the columns represent the power consumed in each hourly period. One of the variables in the COP model is PH. It is an array that represents the power configuration that the customer must hire for

each hourly period. The data model (DM) contains the input values (IN) required by the COP (T is assigned to $Tariff$ and C is assigned to $Consumptions$). It also includes the output values (OUT) produced by the COP (in this case, PH is assigned to $PowerToHire$ and TI is mapped to $TotalInvoince$), as well as other attributes (OT) that will be useful for other purposes. The attributes of the dataset (DS) must be properly transformed according to the data model (DM). In this scenario, the use of Data Preparation techniques is ideal. For example, the nested structure *consumption* in the dataset (DS) must be mapped to *Consumptions* in the data model (DM). It implies a complex Data Transformation, since it requires to transform a complex data structure. This motivates the use of the Data Chameleon DSL to perform this transformation.

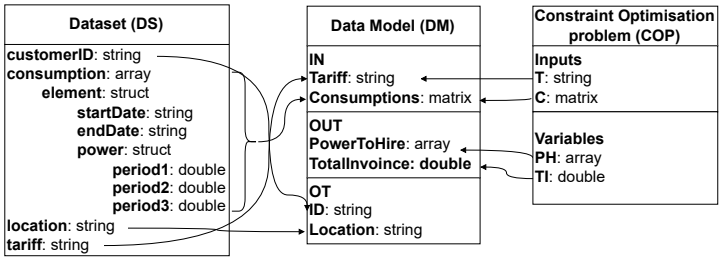


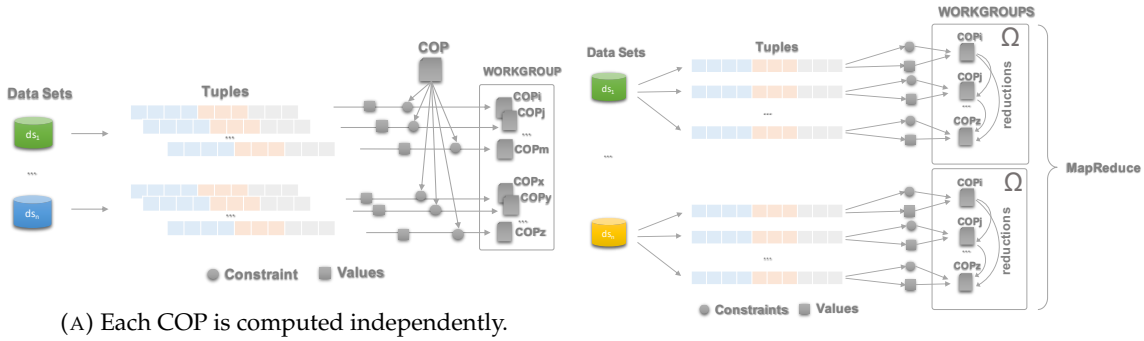
FIGURE 2.10: Example of data mapping between a dataset, a data model and a COP.

As mentioned above, our proposal supports querying operators to make queries to DS^{output} through the data model (DM). Depending on the type of query, the execution of the COPDDs can be optimised. First, we describe the operators that are supported:

- Selection (σ). It enables selecting specific rows in DS^{output}. This query can be performed before or after the COPDDs are computed.
- Projection ($\prod$). It allows the selection of specific attributes within DS^{output}. If the attributes selected in the projection are within the OUT group, then the implied COPDDs must be executed previously.
- Aggregation (Ω). This operation implies performing a calculation that involves a group of records. In our proposal, the following aggregator operators are supported: SUM (it sums the values of a specific attribute within each group of records), $COUNT$ (it counts the number of records within each group), AVG (it calculates the average value of a specific attribute within each group of records), MAX and MIN (these are employed to obtain the maximum and minimum values of a specific attribute within each group of records). This operation can be applied to attributes IN, OUT and OT. If an attribute within OUT is involved in the aggregation, then the COPDDs must be executed.

The closure property of the relational algebra allow to combine these operators. These also enable to optimise the execution of the COPDDs in twofold. First, depending on the query, not all COPs in the dataset must be computed. For example, if a selection operation is performed, only the COPs related to the selected rows are computed. Second, if aggregation operators are employed, in certain cases the COPs can be grouped in different *workgroups*. In

our proposal, a workgroup is a group of COPs that can be computed together taking advantage of the information provided by the computation of COPs within the same workgroup. Therefore, our proposal provides two modes of execution. Figure 2.11a represents the execution mode in which each record generates a COP, each COP being computed independently and distributed among the nodes in the distributed environment. On the other hand, Figure 2.11b illustrates the execution mode when an aggregation operator is performed. In this scenario, the COPs are grouped into workgroups. Then, each workgroup is computed at a specific node of the distributed environment. The optimisation to apply depends on the aggregation operator. For example, with respect to the aforementioned real-world scenario, an aggregation query could be: *Obtain, for each location, the customer who would pay the least amount when contracting the tariff configuration recommended by the COP*. This query implies grouping the records by the *Location* attribute of the data model, and using the *MIN* aggregation operator on the *TotalInvoice* attribute of the data model. Since this attribute is of type *OUT*, its value depends on the COP result. An optimisation that can be applied here is to use the value for *TotalInvoice* obtained for each COP within the same workgroup to bound the domain of this variable to the rest of the COPs within the same workgroup. This leads to a reduction of the search space on each COP and, therefore, could improve the computation time.



(A) Each COP is computed independently.

(B) COPs are computed in workgroups.

FIGURE 2.11: Two different COP computation modes.

This proposal is supported by FABIOLA (FAst BIg cOnstraint LAboratory) [6], a tool with a user interface that implements the different stages of our proposal. This tool is described in our work [90]. Next, we summarise the components of it.

- **FABIOLA Metastore**. It stores metadata comprising the definition of COPs, data models, data mappings, and the results of COPs. It is implemented on MongoDB [47] and Hadoop HDFS [39].
- **FABIOLA Nodes**. This is the cluster of nodes that is responsible for generating and computing the COPs on a distributed basis. This cluster is implemented in Apache Spark [7].
- **FABIOLA UI**. This is the user interface that implements the core functionalities of our proposal. The components of the interface enables to define and execute the COPDDs.

[6]FABIOLA for Spark: `https://github.com/IDEA-Research-Group/fabiola-spark-jobs`, FABIOLA user interface: `https://github.com/IDEA-Research-Group/fabiola-gui-core`

  - *Data importer*. It enables importing the datasets, supporting several types of data sources.
  - *Repository of data mappings*. It enables the creation, review, and editing of data mappings between a specific dataset and a COP definition.
  - *Repository of problem configuration*. This component allows to model, review, and edit COPs. The current version of FABIOLA supports the modelling of COPs with the choco-solver library [68] by using the Scala programming language [80].

- **FABIOLA Dashboard**. It is a dashboard where the user can use several reporting and querying components.

The case study we presented in [90] was successfully solved. We compared parallel execution to sequential execution, and we also tested the scalability of the proposal as the number of COPs to solve increased. We managed to significantly reduce the computation time with respect to the sequential computation of the COPs (about 65% on average).

2.4.2 Computing and distributing big complex optimisation problems applied to conformance checking

In certain cases, the optimisation problem is so large that it cannot be easily distributed. In these cases, the context of the problem must be studied carefully to analyse whether it can be divided and distributed. In this thesis, we focus on the particular case: The conformance checking. The conformance checking discipline is a process mining technique that enables to discover deviation and abnormalities in business process models. Therefore, this technique is of paramount importance to assert the quality of a business process. Conformance checking enables to relate business process models and observed behaviours (i.e., event logs from that process model). The core of this technique is based on a highly complex optimisation problem: The *alignment problem*. In summary, this technique consists of computing an end-to-end model run that more closely resembles a specific trace within the event log. The alignment problem is, in essence, an optimisation problem in which a search space must be explored to find optimal values. Since the complexity of this problem is exponential [3], the computation time does not scale well in the cases where (i) the process model becomes more complex and (ii) the amount of event logs increases.

In order to face highly complex business processes, decompositional techniques have been widely employed. These techniques allow us to divide the model into different parts. In this way, the number of alignment problems increases, but these tend to be less complex. In our proposal, we present a new decomposition technique: Horizontal decomposition, which seeks to reduce the complexity of complex process models with high cyclicity. Unlike the traditional vertical decomposition that breaks both traces and process models into different fragments in order to minimise the search space of each alignment problem, the horizontal decomposition generates end-to-end cuts of the process model. Cuts are generated by *acyclic covers* (i.e., end-to-end runs without cycles). In our proposal, these cuts, which are partial representations of the original process model, are called *partial models*, being less complex than the original

process model due to the lack of cycles. Once a process model is decomposed, each partial model can be combined with the traces of the event log in order to build alignment problems. Since the partial models are less complex, the search space of the alignment problems is reduced, and hence the computation time is dramatically reduced. Once the alignment problems derived from the partial models have been solved, the solutions can be combined to find the alignment value. Basically, the solution is the alignment problem that gives the optimal alignment value.

The second challenge is related to computing alignment problems with large amounts of event logs. To face this challenge, our proposal distributes the traces of the event logs along the nodes of a Big Data cluster. Since the alignment problem seeks to find the best alignment for each trace, each trace of the event log can be combined with the process model to generate an alignment problem. Our proposal goes one step further and allows each trace of the event log to be combined with each partial model. The following steps indicate how this works:

1. The event log is distributed, producing a total of n traces. Let $Tr = \langle tr_0, tr_1 \dots tr_{n-1} \rangle$ be the set of traces.

2. The process model is horizontally decomposed, producing a total of m partial models. Let $Pm = \langle pm_0, pm_1 \dots pm_{m-1} \rangle$ be the set of partial models.

3. The set of traces and the set of partial models are combined through the Cartesian product. This produces a set $AP = (Tr \times Pm)$. AP contains all existing combinations between the elements of Tr and Pm. We say that $ap = \langle tr_i, pm_j \rangle, ap \in AP$ is an *alignment subproblem* (i.e., each element of AP is called an alignment subproblem).

4. AP is distributed along the nodes of the cluster according to the distribution criteria selected by the user. Figure 2.12 depicts this distribution process. The column *Input* represents AP. Then, it is distributed into partitions of alignment problems (see the *Partitions* column). Each partition contains a number of alignment problems that are computed at a specific node of the cluster, being isolated from the rest of the partitions.

5. Next, a *MapReduce* [27] operation is initiated. The Map phase is now described (see column *Map* in Figure 2.12). Within each partition, the alignment subproblems are grouped by trace. Next, an estimation is calculated for the alignment subproblem. Although the estimation function can be specified by the user, we propose an algorithm that calculates a lower bound for the alignment value of each $ap = \langle tr_i, pm_j \rangle$, obtaining the lowest value that the alignment can take. Next, each group is sorted from lowest to highest estimate. The alignment subproblems are then computed. If the result of one of these computations produces a better alignment than the estimation for the same trace, then the computation of alignment subproblem is stopped for that group. At the end of this phase, there is an alignment value for each single trace within each partition.

6. Finally, during the Reduce phase, the partitions are combined. For each trace, the best alignment value found is returned.

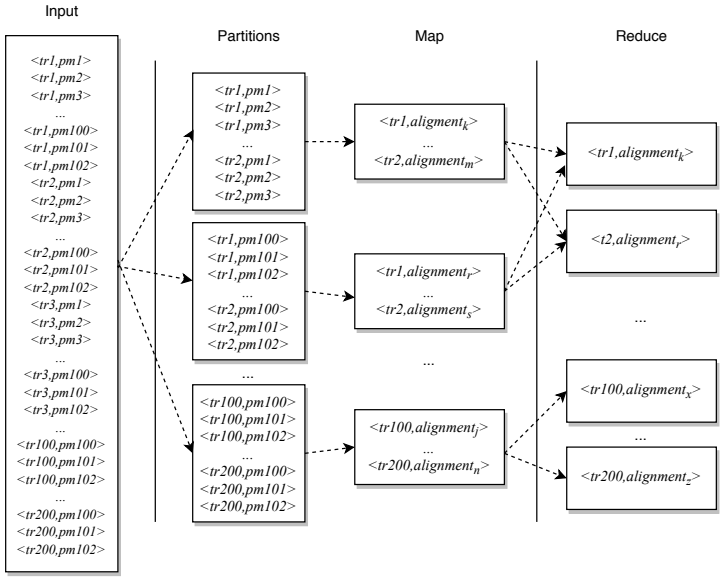


FIGURE 2.12: Schema of the distribution and computation of the alignment problems through the Map Reduce paradigm.

Our proposal is agnostic to the algorithm employed to compute the alignments. In our benchmark, we use a well-known algorithm based on A* [18] implemented in the PM4Py [66] Python library. We also propose an innovative approach based on the constraint optimisation paradigm: Our work includes a Constraint Optimisation Problem model that enables one to solve conformance checking problems defined as a set of variables, constraints, and an objective function.

We developed a tool, CC4Spark[7] [82], that enables one to generate and distribute conformance checking problems. It takes as input a process model or a decomposed process model and event logs. It applies the methodology explained above and returns the alignment values for each trace. This tool is based on PySpark (the Python version of Apache Spark [7]) and the PM4Py library. While Spark is employed to facilitate the distribution of event logs in Big Data clusters, PM4Py facilitates the serialisation of process models and event logs. It also provides a set of algorithms to compute alignment problems.

The proposal was tested with five well-known datasets in the conformance checking community due to their complexity. The performance of CC4Spark with the A* algorithm and the proposal based on the constraint optimisation paradigm was compared with the traditional standalone approach with the A* algorithm. CC4Spark managed the best results in terms of computational time in four of the five datasets.

[7]CC4Spark: `https://github.com/IDEA-Research-Group/conformancechecking4spark`

2.5 Overall picture and Summary

This Chapter presented the results of this thesis. These are focused on improving Data Preparation, Data Quality and Data Analysis in those Big Data pipelines that are intended to compute optimisation problems in cases where the input data is distributed and meets the characteristics of Big Data (i.e., volume, variety, and velocity), and/or the problems are too large and complex, requiring a special treatment. Moreover, the type of problems that we are facing require (i) processing complex data structures, and (ii) extracting event logs in a standard format from semi-structure data. The optimisation techniques that we use are also sensitive to bad-quality data, since this type of data leads to bad results. Therefore, the proposals in this thesis are intended to enrich the techniques that are available at different stages of Big Data pipelines. The tools developed are also designed to be used in Big Data pipelines through the Apache Spark [7] data processing engine.

Next, we present two Big Data pipelines that have been employed to solve several case studies. The first is shown in Figure 2.13 and includes the use cases **UC-DT-1**, **UC-DQ-1**, and **UC-OP-1**. The ultimate objective of this pipeline is to support the Data Analysis in the use case **UC-OP-1**. The FABIOLA solution (Section 2.4.1) must be used to calculate large amounts of individual COPs. In this particular case, the data are acquired from one of the three electricity distribution companies. The data contains information about the consumption of each customer. Consumption is given within a nested data structure. Next, in the Data Preparation stage, we employ the Data Chameleon DSL [83] (Section 2.2.1) to transform the source data schema into the format required by the Data Model of the COP. Subsequently, the Data Quality of each record is assessed. This is done using DMN4Spark applying the DMN4DQ methodology [89] (Section 2.3.1). The purpose is to filter out the data records that are non-usable. In this case, we wanted to avoid data records with values *null* and outside the range. Then, the Data Filtering activity forks the data flow so that the data that have been qualified as usable go to the next stage of the pipeline (i.e., Data Analysis), while the non-usable data are stored in the data storage system. In this way, non-usable data can be diagnosed and repaired using DMN4DQ+ (Section 2.3.2). Finally, FABIOLA is employed as a Data Analysis tool. It receives the records from the dataset that have been qualified as usable and collects the Data Model and COP model from the data storage system.

The second Big Data pipeline is presented in Figure 2.14, which includes the use cases **UC-DT-2**, **UC-DQ-1**, and **UC-OP-2**. This is orientated towards the process mining paradigm, since the Data Analysis stage intends to employ conformance checking techniques to compute the alignment between a process model and the event logs of such a process. Conformance checking involves the computation of optimisation problems with large search spaces. In this case, we use CC4Spark [85]. This proposal has been summarised in Section 2.4.2. The case study presented in Figure 2.14 was presented in [82], and is related to the milk manufacturing industry. The purpose of this pipeline is to discover potential errors and defects in this process. In the data acquisition activity, raw event logs from the process are collected. The Data Quality of each record is then evaluated with DMN4Spark, following the DMN4DQ methodology [89] (Section 2.3.1). In this case, incomplete and inaccurate data is penalised. As in

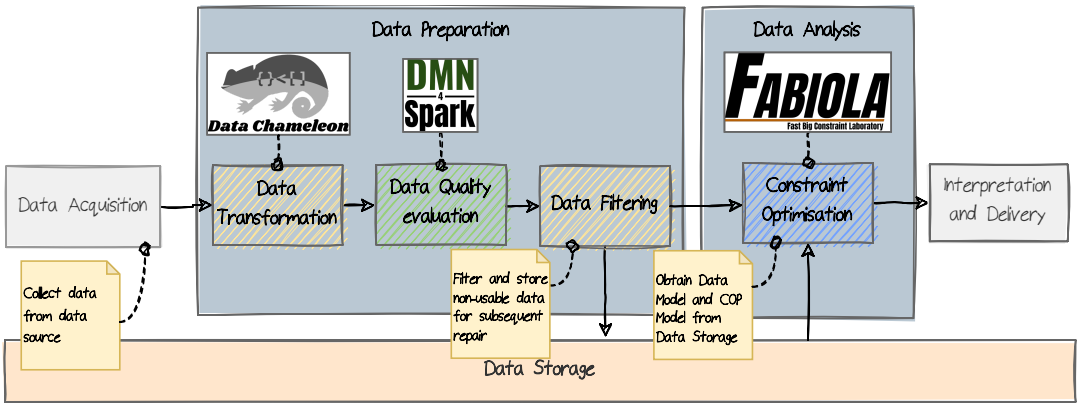


FIGURE 2.13: A Big Data pipeline used in our case studies to compute individual Constraint Optimisation Problems.

the previous pipeline, the Data Filtering activity separates usable data from non-usable data. While usable data are sent to the next Data Preparation activity, non-usable data are stored in the data storage system for later repair with DMN4DQ+ (Section 2.3.2). The last Data Preparation activity obtains the usable data and extracts the event logs so that they follow the XES standard [44]. This operation is carried out through the ELE DSL [87] [86] (Section 2.2.2). Once the event logs are accurately formatted, these are sent to the Data Analysis stage, where CC4Spark is employed to distribute and compute the alignment problems. The decomposed process model is obtained from the data storage system.

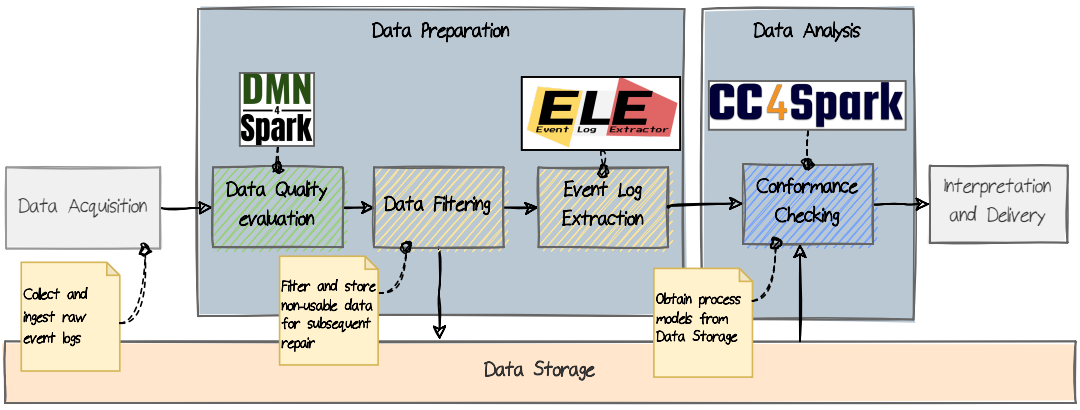


FIGURE 2.14: A Big Data pipeline used in our case studies to compute big complex conformance checking problems.

The next Chapter discusses the results that have been obtained.

Chapter 3

Discussion and Publications

This Chapter discuses the solutions that have been developed to satisfy the objectives of this thesis. It also presents and discusses the contributions that comprise and support this thesis. Therefore, Section 3.1 discusses the results of this thesis, first by presenting the contributions (Section 3.1.1), and then by discussing the fulfilment of the objectives of this thesis (Section 3.1.2). Finally, the main publications are shown in Section 3.2.

3.1 Discussion of the Results

This section discusses the results and contributions derived from this thesis. Table 3.1 outlines the results that have been produced for each objective traced for this thesis, classified by scope (i.e., the different stages of the Big Data pipeline that are covered in this thesis), the publications produced for each objective and the tool or tools that support each proposal.

Scope	**OBJ 1 - Data Preparation**		**OBJ 2 - Data Quality**		**OBJ 3 - Data Analysis**	
Objectives	OBJ 1.1	OBJ 1.2	OBJ 2.1	OBJ 2.2	OBJ 3.1	OBJ 3.2
Publications	ICIS'19 [83]	BPM'19 Industrial Forum [87] BPM Cases Vol. 2 [86]	DEC2H'19 [84] DSS [89]	DSS[1]	JCOS [90]	Inf.Syst. [85] BPM'21 Tools & Resources [82]
Tools	Data Chameleon DSL	Event Log Extractor DSL	DMN4Spark DMN4DQ+		FABIOLA	CC4Spark

TABLE 3.1: Summary of the objectives and results of this thesis.

3.1.1 Contributions

The contributions are divided into two groups. The first is composed of a compendium of articles from this thesis. The second group consists of supporting articles that support the objectives of this thesis but are not part of the compendium. Next, the contributions that form the **compendium of articles** are listed:

- Á. Valencia-Parra, Á.J. Varela-Vaca, M.T. Gómez-López, P. Ceravolo (2019). **CHAMALEON: Framework to improve Data Wrangling with Complex Data**. *In International*

[1]This contribution has been submited to the Decision Support Systems journal and is currently under revision.

Conference on Information Systems (ICIS 2019) Proceedings. 16 [83] **Rank**: SCIE RANK GGS' 21 Class 2.

- Á. Valencia-Parra, L. Parody, Á.J. Varela-Vaca, I. Caballero, M.T. Gómez-López (2021). **DMN4DQ: When data quality meets DMN**. *In Decision Support Systems (Vol. 141, p. 113450). Elsevier BV.* [89]. **Rank**: Q1 (JCR'20 5.795).

- Á. Valencia-Parra, Á.J. Varela-Vaca, L. Parody, M.T. Gómez-López (2020). **Unleashing Constraint Optimisation Problem solving in Big Data environments**. *In Journal of Computational Science (Vol. 45, p. 101180). Elsevier BV.* [90]. **Rank**: Q2 (JCR'20 3.976).

- Á. Valencia-Parra, Á.J. Varela-Vaca, M.T. Gómez-López, J. Carmona, R. Bergenthum (2021). **Empowering conformance checking using Big Data through horizontal decomposition**. *In Information Systems (Vol. 99, p. 101731). Elsevier BV.* [85]. **Rank**: Q3 (JCR'20 2.309).

The following list illustrates the contributions that support this thesis:

- Á. Valencia-Parra, B. Ramos-Gutiérrez, Á.J. Varela-Vaca, M.T. Gómez-López, A. García Bernal (2021). **Enabling Process Mining in Airbus Manufacturing**. *In Business Process Management Cases Vol. 2 (pp. 125-138). Springer Berlin Heidelberg.* [86].

- Á. Valencia-Parra, L. Parody, Á. J. Varela-Vaca, I. Caballero, M.T. Gómez-López (2019). **DMN for Data Quality Measurement and Assessment**. *In Business Process Management Workshops (pp. 362–374). Springer International Publishing.* [84].

- Á. Valencia-Parra, Á.J. Varela-Vaca, M.T. Gómez-López, J. Carmona (2021). **CC4Spark: Distributing Event Logs and big complex Conformance Checking problems**. *In Proceedings of the Best Dissertation Award, Doctoral Consortium, and Demonstration and Resources Track at BPM 2021 co-located with 19th International Conference on Business Process Management (BPM 2021). Vol. 2973 (pp. 136-140) CEUR-WS.org.* [82].

- Á. Valencia-Parra, L. Parody, Á.J. Varela-Vaca, I. Caballero, M.T. Gómez-López (2021). **Optimising Data Reparation to enhance Data Usability**. *In Decision Support Systems. Elsevier BV.* [2]

3.1.2 Discussion

OBJ 1.1. Develop a DSL supported by a framework to transform complex data

The solution to this objective is the Data Chameleon framework (presented in Section 2.2), and the Data Chameleon DSL (described in Section 2.2.1). This proposal was presented in *International Conference on Information Systems 2019* (ICIS 2019) [83]. The paper can be found in Section 3.2.1. The conference's SCIE rank GGS'21 is *Class 2*. This work was carried out in collaboration with Professor Paolo Ceravolo of the SESAR Lab Research Group [3] from the

[2] This contribution has been submited to the Decision Support Systems journal and is currently under revision.
[3] SESAR Lab Research Group: `https://sesar.di.unimi.it/`

University of Milan. Paolo is expert in the area of Big Data pipelines, data lakes and process mining.

The proposal was demonstrated to work properly in two real-world case studies. The first one was focused on the transformation of complex data for a Big Data pipeline that was intended to compute Constraint Optimisation Problems (COPs). Data were provided by three different datasets from three electricity distribution companies in Spain. These companies required the use of COPs to analyse their data and, therefore, to facilitate the decision-making about the optimal tariff configurations for their customers. Data Chameleon was successfully used as a Data Preparation tool, since it was able to transform these datasets according to the data model required by the optimisation problems. On the other hand, the second case study, which has been discussed in Section 2.2.1, required transforming the aircraft assembly and testing log data in Airbus.

The proposal aimed to simplify the Data Transformation process with complex structures, since traditional tools are focused on table-based data structures, requiring intermediate abstract operations to flatten nested structures and operate at the table level. Data Chameleon eliminates this need, making all transformations directly have their intended effect on the target schema.

In our study, we solved both case studies with Data Chameleon DSL and a leading Data Wrangling tool (Trifacta). We found that with the latter tool, more than 50% were intermediate operations derived from flattening and grouping operations. With Data Chameleon, none of these types of operations was necessary. We also tested the performance and scalability of the Data Chameleon ecosystem. For this purpose, we carried out different benchmarks by scaling the datasets in twofold ways: (A) by increasing the number of records of the datasets, and (B) by augmenting the size of the nested structures. We found that Data Chameleon scales correctly since the execution time in both cases evolves in a similar way.

OBJ 1.2. Develop a DSL to support the extraction of event logs from semi-structured data

The solution we propose to accomplish this objective is the Event Log Extractor (ELE) DSL. This language, which is based on the Data Chameleon framework, was briefly described in Section 2.2.2. This solution was presented in *Business Process Management Industry Forum 2019* [87]. Later, a chapter was published in the *BPM Cases Volume 2* [86] book. This work was carried out in close collaboration with the Airbus company in a project that sought to discover business processes in the aircraft manufacturing and testing process, as well as diagnose anomalies and deviations. As huge amounts of logs from these processes were required to be processed, and since these logs had complex nested structures, advanced Data Preparation techniques were required. Bearing this challenge in mind, during the development of this thesis, the potential of the Data Chameleon framework was used to develop a new DSL that would allow event logs to be transformed and formatted to the XES standard format. The transformation of raw event logs into the XES format is a crucial step in the analysis of the logs, since the process mining algorithms work with this specific format. Thanks to the ELE DSL, it was possible to properly transform the Airbus event logs. In the work that we presented, several transformations were carried out in order to analyse the event logs from

different points of view. The results of the analyses allowed the company to find hidden processes and anomalies.

OBJ 2.1. Develop a methodology to model Data Quality requirements in a context-aware basis

This objective is achieved with DMN4DQ, a methodology that proposes a systematic procedure to model the Data Quality requirements hierarchically. Section 2.3.1 presents this methodology. The solution was published in the *Decision Support Systems* journal [89]. The journal rank is Q1 (JCR'20 5.795). Professor Ismael Caballero, from the University of Castilla-La Mancha, collaborated in the development of this work as an expert in the field of Data Quality.

The methodology enables us to create Data Quality models through business rules. These are modelled by following a hierarchical basis and taking into account the context in which data are used. DMN is the decision modelling framework that supports the proposal.

DMN4DQ was successfully evaluated in a real-world case study. The context of use of the data consisted of a catalogue in which third-party cloud server instances are offered, mainly from Amazon Web Services. In this context, the negative impact that poor Data Quality could have in terms of poor reputation and failures in contracting services should be taken into account. DMN4DQ allowed weighting the importance of three dimensions of Data Quality (in this case, accuracy, consistency, and completeness). In addition, it allowed for the establishment of validation rules for the different data attributes, as well as the importance of complying with each of these rules. It also made it easier to weigh the importance of each Data Quality dimension in the final assessment of data usability. The Data Quality model for this case study is made up of 18 DMN tables, with a total of 53 decision rules.

A tool (DMN4Spark) was also developed. It allowed the application of the *Camunda DMN* decision rule engine to large volumes of data using the Apache Spark data processing framework. In this way, this methodology can be applied in Big Data environments, supporting the Data Quality activity within the Big Data pipelines. This tool was used to solve the case study, whose dataset contained $1,048,571$ records. For each record, a recommendation on its usability was produced.

The DMN4DQ methodology has also been used in an IoT scenario in a work [40] that was carried out in collaboration with the Universidad of Almería. In that work, DMN4DQ is used to support the data curation process. The scenario is based on the agriculture sector. Data are produced by a series of sensors placed on several farms. In this case, DMN4DQ made it possible to model quality rules that allowed data from sensors that did not meet the appropriate standards to be discarded for later analysis.

OBJ 2.2. Develop a framework to facilitate the detection of the root causes of poor quality data, as well as data repair

This objective has been addressed using the DMN4DQ+ solution, which was discussed in Section 2.3.2. The paper related to this solution has been submitted to *Decision Support Systems*.

As in the case of DMN4DQ, Professor Ismael Caballero, an expert in the area of Data Quality, has collaborated in the development of the proposal at the methodological level.

DMN4DQ+ is an extension of DMN4DQ that adds mechanisms to facilitate Data Quality characterisation, finding the root causes of poor Data Quality, and decision-making about data cleaning operations to be carried out to improve the usability of the data.

The solution was applied to the case study presented in [89], and that has been discussed in the previous Section. In this case, the different usability profiles that exist in the dataset were calculated. This allowed us to analyse which Data Quality problems affected most of the tuples. A total of 29 usability profiles were identified, of which 22 were marked as "do not use". A set of 27 candidate corrective actions was used. In the benchmark, we sought to repair all usability profiles to find the subset of corrective actions with the lowest cost for each of them. Therefore, a COP was run for each usability profile. The execution time is reasonable, as 21 of the 22 COPs were resolved in less than 6 seconds. However, one of them took 222 seconds. We managed to explain this fact by finding a correlation between the number of Data Quality problems in a usability profile and the execution time of the COP, so that the more Data Quality problems a usability profile has, the larger the search space of the COP will be, and therefore the longer the execution time. On the other hand, we performed a study on the cost savings of improving the usability of each usability profile. We carried out this study by comparing the cost of applying the corrective actions selected by the COP in each usability profile with the total cost of applying all the corrective actions that could be applicable to each usability profile. While in 7 of the 22 usability profiles the cost savings were 0, in the rest we obtained significant improvements in terms of cost savings.

OBJ 3.1. Develop a framework to model, generate, and solve Constraint Optimisation Problems with Distributed Data

The solution to this objective is FABIOLA (described in Section 2.4.1). This proposal has been published in *Journal of Computational Science*. The journal's rank is Q2 (JCR'20 3.976). In this paper, the modelling, generation, and computation of Constraint Optimisation Problems (COPs) are formalised. A methodology and a tool (FABIOLA) are proposed to optimise the computation of COPs in Big Data environments.

The case study employed to test this proposal focusses on three Spanish electricity distribution companies. They require generating, for each customer, a tariff configuration that allows reducing the cost of the electricity bill, taking into account the previous consumption of each customer. For this case study to be solved, the COP model was first generated. This contained the input parameters required to compute the COP and the variables, constraints, and objective function. Second, the data model was generated. This indicates the input parameters required by the COP, the output parameters embedded from the COP, and other data attributes that were useful for performing queries. Next, the attributes of each dataset must be mapped to the data model. For this purpose, Data Preparation tasks must be performed. We employed Data Chameleon to transform the data schema of each dataset according to the COP data model, and DMN4DQ to filter those records that might cause failures in the computation of the COPs due to poor-quality data.

Regarding the benchmark, the aim was to demonstrate the scalability of the proposal as the number of COPs to be solved increased. For this, an asymptotic analysis was performed. For each dataset, the computation time of each of them was sampled from $5,000,000$ to $25,000,000$ COPs. The performance of the FABIOLA solution (the COPs are distributed and resolved in parallel) was compared with a sequential execution. The asymptotic study revealed that the complexity of the sequential execution is linear (i.e., $\mathcal{O}(n)$), while the execution using FABIOLA (in parallel) presented a sublinear complexity, three times lower than the sequential execution (i.e., $\mathcal{O}(\frac{n}{3})$). Clarify that the cluster in which FABIOLA was launched had 4 nodes.

A second benchmark was performed to check the impact when aggregation operators are introduced (in this case, grouping operators). In summary, the application of these operators did not improve performance in terms of time due to the impact of grouping operations. However, if we compare the COP resolution stage, we observe an improvement in two of the three datasets (i.e., a 9.7% of improvement in one of them and a 41.6% in the other). We concluded that the impact of the optimisation of aggregation operators depends on the nature of the data.

OBJ 3.2. Develop a solution to compute big complex optimisation problems in Big Data environments applied to conformance checking techniques

This objective is achieved through the proposal that we published in the *Information Systems* journal [85]. The journal's rank is Q3 (JCR'20 2.309). This work was made in collaboration with Professors Josep Carmona and Robin Bergenthum, both of whom are experts in the process mining and conformance checking fields. The proposal resulted in a tool (*CC4Spark*), which was presented in *Business Process Management Demonstrations & Resources 2021* [82].

The proposal is intended to facilitate the computation of complex conformance checking problems in those scenarios where one or both conditions are met: (i) the business process model is too complex, and (ii) the dataset containing the event logs is too large. Usually in these cases traditional Conformance Checking solutions fail to find a solution in a reasonable time, as we demonstrated in our study.

In order to validate our proposal, we employed five different datasets that are known by the conformance checking community because of their complexity. Thanks to our proposal, the complex Petri nets could be broken down into smaller units. On the other hand, the event logs could be distributed among the nodes of a Big Data cluster. First, we compare the sequential computation of the conformance checking algorithm based on A* with the parallel computation that we propose. We also tested the parallel computation of the solution based on COPs. Regarding parallel execution, several distribution configurations were tested to find the best balance between the number of partitions and the number of partial problems within each partition. We got a very significant improvement by parallelising the computation of conformance checking problems. In summary, the sequential A* algorithm showed better results in one of the five datasets. On the other hand, the parallel A* algorithm using our proposal achieved the best results in two of the five datasets. Finally, the algorithm based on parallel COPs that we propose showed better results in two of the five datasets. With respect to the results, the A* algorithm, both sequentially and in parallel, was able to produce optimal

solutions for each dataset. However, the COP-based solution was not always able to return an optimal solution due to the setting of timeouts. However, the percentage of optimal solutions was higher than 80%.

Our solution, CC4Spark, was also applied in the case study presented in [82]. This scenario consists of a milk manufacturing process that describes how milk cans are processed through the production chain. Event logs were produced by IoT sensors located within each machine, collecting data on temperature and pressure. This case study was successfully solved and managed to monitor the whole process and detect abnormalities in the final products.

3.2 Publications

3.2.1 CHAMALEON: Framework to improve Data Wrangling with Complex Data

Published in the International Conference on Information Systems (ICIS 2019), Munich, Germany. December 2019.

- ***Authors**: Álvaro Valencia-Parra, Ángel Jesús Verala-Vaca, María Teresa Gómez-López, Paolo Ceravolo.*
- ***URL**: `https://aisel.aisnet.org/icis2019/data_science/data_science/16`.*
- ***Rating**: GGS Class 2.*

CHAMALEON: Framework to improve Data Wrangling with Complex Data

Completed Research Paper

Álvaro Valencia-Parra
Universidad de Sevilla
Avda. Reina Mercedes S/N 41012
Sevilla - SPAIN
avalencia@us.es

Ángel Jesús Varela-Vaca
Universidad de Sevilla
Avda. Reina Mercedes S/N 41012
Sevilla - SPAIN
ajvarela@us.es

María Teresa Gómez-López
Universidad de Sevilla
Avda. Reina Mercedes S/N 41012
Sevilla - SPAIN
maytegomez@us.es

Paolo Ceravolo
Università degli Studi di Milano
Via Giovanni Celoria, 18, 20133 Milano – ITALY
Paolo.Ceravolo@unimi.it

Abstract

Data transformation and schema conciliation are relevant topics in Industry due to the incorporation of data-intensive business processes in organizations. As the amount of data sources increases, the complexity of such data increases as well, leading to complex and nested data schemata. Nowadays, novel approaches are being employed in academia and Industry to assist non-expert users in transforming, integrating, and improving the quality of datasets (i.e., data wrangling). However, there is a lack of support for transforming semi-structured complex data. This article makes a state-of-the-art by identifying and analyzing the most relevant solutions that can be found in academia and Industry to transform this type of data. In addition, we propose a Domain-Specific Language (DSL) to support the transformation of complex data as a first approach to enhance data wrangling processes. We also develop a framework to implement the DSL and evaluate it in a real-world case study.

Keywords: Data Wrangling, Complex Data, Data Transformation, Semi-structured Data, Data Preparation

Introduction

The continuous technological advances in Industry are leading to data-driven business processes. These changes (Gerbert et al. 2015) are motivated by the concept of Industry 4.0, whose objective is to improve their production processes by means of Cyber-Physical Systems. These systems enable companies to capture real-time data on any aspect related to these productive processes. On the other hand, Industry 4.0 promotes companies to a broader integration with their external environment. Therefore, the necessity of integration of internal data with data from external sources (e.g., data from other organizations, open data, social network data) arises (Obitko and Jirkovský 2015). The ultimate objective is to process this data in order to discover knowledge, improve the decision making, and optimize production processes. Big Data technologies are an essential part in this industrial context (Gerbert et al. 2015) since data to be processed fulfill the three Big Data dimensions (a.k.a., the three V's) (Lee 2017): volume (they are massively generated), velocity (creation rates increase as Cyber-Physical Systems and IoT are included in production

processes), and variety (data become more heterogeneous as the amount of data sources increase). This new context has promoted the creation of new solutions adapted to the complexity of data.

Data complexity and heterogeneity are a prominent challenge. When the amount of data and the creation rate increase drastically, data heterogeneity tends to rise as well, resulting in non-structured data models with nested and complex schemata (hereinafter, complex data). Derived from the data complexity, data integration becomes also in a prominent challenge of Big Data management (Ceravolo et al. 2018; Jin et al. 2017; Stefanowski et al. 2017), since it involves combining diversified sources in a unified view supporting data analytic or reporting procedures (Dong and Srivastava 2015). Data integration requires, therefore, several transformations to be made, changing data values and structure but keeping at the same time the validity and consistency of data or event enhancing their value. For this reason, data transformation, as a part of the data preparation process, is considered the most time-consuming stage of data analytics (Guo et al. 2011).

In traditional data warehousing, this process has been extensively analyzed, proposing integration tools for data transformation by using the Extract-Transform-Load (ETL) approaches. Regarding the complex data, various query languages can be found in industry (Beyer et al. 2011; Florescu and Fourny 2013) to facilitate the transformation, integration and querying of data sources with complex data in ETL processes. However, with the emergence of the Big Data paradigm, actors focusing on the commoditization of Big Data technologies are addressing the fast roll-out of Big Data pipelines proposing visual data flow orchestrators (Milutinovic et al. 2017) and catalogs of congruent services (Ardagna et al. 2018). The aim is the introduction of the as-a-service approach in Big Data technologies, supporting the composition of services in an easy way and offering, at the same time, a guarantee about the consistency of the proposed compositions (Ardagna et al. 2018). In this sense, data wrangling has become one of the most employed techniques to facilitate the transformation and mapping of data from a raw format into the format required by data analysis processes in Big Data context. The current trend is to provide self-service data preparation (Hellerstein et al. 2018). It points at easing the data preparation process for non-expert users through data profiling and the automation of the tasks. Therefore, this assistance comes along with user-friendly Domain-Specific Languages, user interfaces, and features for data cleaning and data quality improvement.

Nowadays, several data wrangling solutions can be found in Industry. Although some of them are able to work with complex data, they are entirely focused on a table-oriented data model, flattening data into static structures avoiding nested data. It implies that operations must be directly applied over top-level attributes (i.e., columns). This operative inevitably difficulties the transformation of complex structures, requiring nested attributes to be shift to top-level positions. Consequently, (i) the number of operations needed to transform the format and/or schema of a dataset becomes significantly high depending on the depth of the nested attributes and the target schema, and (ii) the definition of the transformation operations becomes non-easy-to-use and anti-intuitive, being far away from the shape of the target schema, and hence, being more error-prone and hindering debugging operations. In this context, the identification of languages for data transformation that support complex operations by a concise syntax is of paramount importance for a flexible handling of data sources (Arputhamary and Arockiam 2015). Moreover, linking these operations to their effects on performances, in relation to the data structure, is crucial to increase the awareness of designers about the effects of their specifications.

Trying to reduce the existing gap in complex data transformation in data wrangling, this paper pursues to cover two main aspects: (i) discuss the approaches in the industry, and the academia to support the transformation of complex data in the data wrangling and self-service data preparation fields, and; (ii) the proposition of a framework, that includes a Domain-Specific Language (DSL), to support the transformation of data with complex schema. This language aims to provide a functional way to enable users to define the target schema along with the transformations needed to reach it from the source schema, minimizing the number of operations and the complexity of the language.

The rest of the paper is structured as follows. First, two real-world case studies are presented to understand the proposal better. Then, our proposal is described, depicting the solution, including a DSL that facilitates the complex data transformation in data wrangling context. The proposed implementation is introduced, before the most relevant references related to data wrangling and complex data are discussed. Next, the related work is discussed, and then we compare our proposal with other data wrangling tools. Finally, some conclusions are drawn.

Case Studies

This section describes two case studies which demonstrate the applicability of our proposal in two different industries that require the transformation of datasets: (A) the transformation of datasets from several electricity wholesales to extract information about the consumption of electricity, and (B) the transformation of datasets taken from several IoT sensors to detect potential deficiencies in aircraft assembly processes.

Case Study A: Formatting datasets from several electricity wholesales

In order to introduce the necessity to facilitate the complex data transformation, we use a real case study based on the integration datasets provided by seven electricity companies that sell energy for private customers in Spain. The electricity wholesales describe consumption data in different formats and using different frequency of meter reading, depending on factors such as the distributor or the tariff hired for each customer. These various formats need to be uniform in order to be processed and analyzed later, such as to create patterns of behavior or to look for the best tariff for each customer (Parody et al. 2017). However, each electricity provider offers information using different nested schemata, depending on some factors, such as the number of months included in the meter reading, number of days, types and tariff. Therefore, all these heterogeneous schemata need to be transformed into a unified one being possible to integrate every dataset in a unified view. The quantity of information, the heterogeneity and the updating of the information, Big Data infrastructure must be used to facilitate the data analysis.

Figure 1 illustrates the scenario where several data sources must be conciliated into a unified format accessible to the final user by means of a set of transformation, each one applicable to a single data source.

As mentioned above, the provided information does not follow the same schema, but they generally share a customer ID, a tariff identifier, the contracted power for each daily billing period, and a list of consumption over a period (e.g., twelve months or more). Each consumption period keeps information on the start and end date for that period, and the power consumption for each daily billing period. Figure 2 shows a possible input schema for the data of the example and its relationships with the target schema.

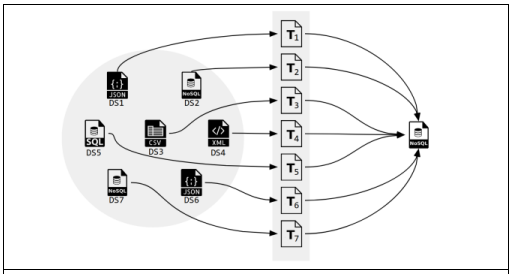


Figure 1. Case Study scenario

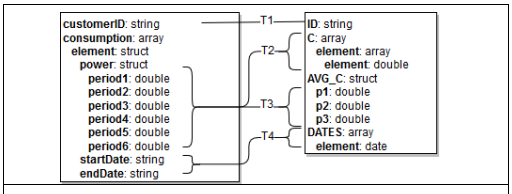


Figure 2. Transformations to convert the source schema (left) into the target schema (right)

Source Schema Description

The source schema is composed of basic and nested attributes. The description of the dataset attributes is given as follows:

- `customerID`. It is a string which identifies a unique customer supply point.
- `consumption`. It is the power consumption over a period, such as twelve months or more. It is an array of data structures. Each element represents a period, and it includes the following information:
  - `power`. It is the power consumption for each daily billing period. It is represented by a data structure with six decimal numeric attributes, each one representing the consumed power for a daily billing period.
  - `startDate`. It is the start date for the billing period represented by this element.
  - `endDate`. It is the end date for the billing period represented by this element.

Transformations and Target Schema

In order to reach the target schema, several transformations must be applied for each electricity supplier. For the example, six transformations are needed as depicted in Figure 2:

- **T1**. `customerID` must be renamed to `ID`. No further transformations are required.
- **T2**. `consumption` is transformed into a matrix `C`, whose rows have three elements that are calculated from the p_i attributes in `power` in accordance with the defined rules by the government.
- **T3**. In the target schema, `AVG_C` is a data structure with three elements: `p1`, `p2`, and `p3`. It represents the average consumed power for each daily billing period. The calculation is carried out by means of the matrix calculated in **T2**.
- **T4**. In the target schema, `DATES` is an array whose attributes are of type `date`. It is calculated by means of the `startDate` attribute in the source schema.

Case Study B: Detecting deficient aircraft

This case study is based on the aeronautic industry. It is a real case study from the aircraft factory of Airbus placed in Seville. The datasets represent the logs of an aircraft production plant. It is about the tests that are performed in the workstations where the aircraft are tested, and the incidents that occurred during these. Each workstation produces a dataset with a different schema and data format but they all have a set of attributes related to the aircraft, the workstation and the incidents occurred during the tests, as detailed in in (Valencia-Parra et al. 2019). It is required to wrangle this data in order to obtain a formatted dataset so that it helps experts to discover potential deficient aircraft.

Source Schema Description

The source schema is composed of the following attributes:

- `accode`. It is a string attribute representing the code of an aircraft.
- `workstation`. It is a string identifying the workstation where the tests have been executed.
- `incidents`. It is an array whose elements represent information about the incidents that have occurred during the test execution. It contains nested structures with the following attributes:
  - `start_date`. It is a string representing the date when the incidence started.
  - `resolution_date`. It is a string representing the date when the incidence was resolved.

Transformations and Target Schema

The following transformations are required in order to reach the target schema:

- **T1** and **T2**. Attributes `accode` and `workstation` are renamed as `aircraft` and `ws`, respectively.

- **T3**. `avg_incidents` is a numeric attribute created by calculating the average value of the array which results from iterating `incidents` and performing the following operation: `resolution_date - start_date`. It represents the average time that the incidents took to be solved.

The objective of these transformations is to produce a dataset that can be processed by an algorithm that detects abnormalities during the test of the aircrafts in the workstations, and thus, to detect potential deficient aircraft.

Our Proposal

This section depicts CHAMELEON: the proposal we have devised to improve the data wrangling processes when dealing with complex data. It consists of framework and a Domain-Specific Language (DSL), whose objective is to link the operations with their effects in the target schema. First, the necessary concepts to understand the proposal are presented. Then, Framework is introduced, following with the proposed DSL. Finally, the case studies are solved by using the proposal.

Related Concepts

Next, the concepts *Data Schema*, *data type*, and *Transformation Function* are defined. These concepts will facilitate the understanding of the DSL that is defined in this section.

Definition. *A Data Schema is a set of attributes, $\{a_1: t_1, a_2: t_2, \ldots, a_n: t_n\}$, identified by a name ($a_i$) and a data type ($t_i$).*

Regarding the data type (t_i), two categories of data types have been identified:

- **Simple type**. It is a data type which represents a single value:
  - `Numeric`. It represents a numeric data type, i.e., Integer, Long, Float, and Double.
  - `String`. It is a sequence of characters.
  - `Boolean`. It is a two-valued data type which represents the truth values.
  - `Date`. It is a set of characters with a specific format that represents an instant of time.
- **Complex type**. It is a composite data type that can be:
  - `Array`. It is a collection of typed attributes identified with a unique numeric index.
  - `Struct`. It is a data type composed of a set of attributes, each one identified by a unique name.

Definition. *A Transformation Function, f_x, is a function that receives an attribute, a_{input}, and returns an attribute, a_{output}, resulting from applying an operation which modifies the value of a_{input}.*

$$f_x: a_{input} \rightarrow a_{output}$$

Framework Modeling

We have developed a framework[1] to implement the DSL so that we can solve the case studies in a real-world environment. The framework has been designed according to the composite design pattern (Riehle et al. 1997). In short, this pattern enables to build complex objects by using simpler ones. It means that an object could be composed of nested objects. Figure 3 depicts a schema of this pattern. As can be seen, the classes `Composite1` and `Composite2` are composed of a set of `Components`, which can be `Composite1`, `Composite2`, or `Leaf`. The latter is called `Leaf` because it is not compounded by any other `Component`. In this pattern, the instances of objects could be represented as a tree structure.

Figure 4 depicts the UML diagram of the transformation framework. As mentioned above, the instances of this model can be represented as a tree structure. In this structure, the leaves are operations that access the attributes, and internal nodes are intended to transform or create new structures. To better understand it,

[1] The implementation can be found in: http://www.idea.us.es/datatransformation/

Figure 5 shows an instance of the transformation T1 exposed in the case study A, and Figure 6 shows its tree representation.

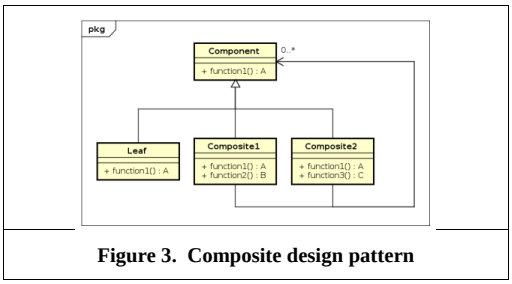


Figure 3. Composite design pattern

In this model, the `Component` is the `Evaluable` interface. An `Evaluable` represents an expression whose main objective is to perform transformation functions on attributes. Two methods can be applied over every `Evaluable` expression (hereinafter, expression): `getValue` and `getDataType`.

- `getValue`. It receives an attribute and returns another attribute as a result of applying a transformation to it.
- `getDataType`. It receives a data type and returns the data type as a result of applying a transformation to it.

These are intended to be the entry-point of the framework. The way these functions work depends on the `Leaf` or the `Composite` components. Next, the leaves of the transformation framework model are listed.

- `Select`. It is meant to select the attribute whose name matches the string `name` from an attribute of type `struct`.
- `Index`. It is meant to select the attribute whose position matches the integer `index` from an attribute of type `array`.

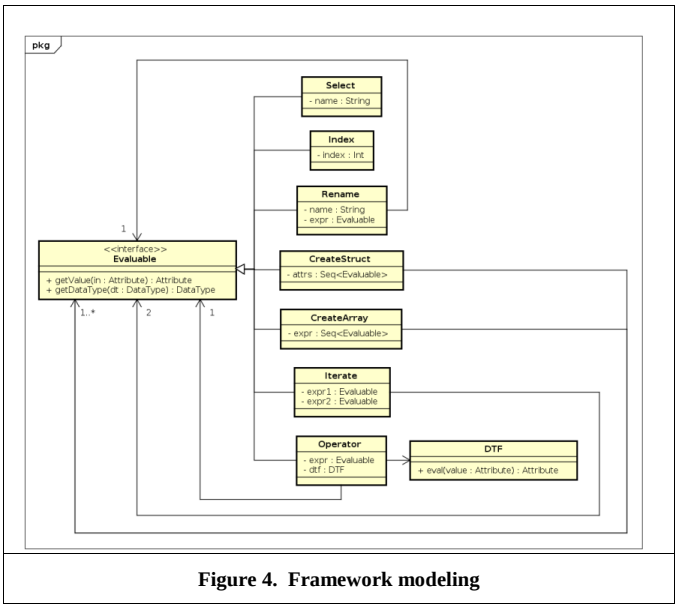


Figure 4. Framework modeling

```
new  Rename("ID", new Select("customerID"))
```

Figure 5. Instance of the model to perform the transformation T1. Code representation

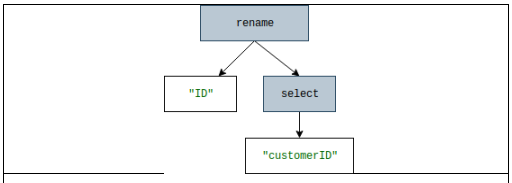


Figure 6. Instance of the model to perform the transformation T1. Tree representation.

Lastly, the `Composites` of the transformation framework model are listed.

- `Rename`. It is meant to transform an `Evaluable` expression (hereinafter, expression) which returns an attribute of any type by replacing its name by the string `name`.
- `CreateStruct`. It is meant to create an attribute of type `struct` from a set of expressions `attrs`.
- `CreateArray`. It is meant to create an attribute of type `array` from a set of expressions `attrs`.
- `Iterate`. It is meant to create an attribute of type `array` resulting from iterating over an expression which returns an attribute of type `array` (`expr1`). An expression (`expr2`) is applied to each element in that `array`.
- `Operator`. It is meant to transform an expression by applying a Data Transformation Function (hereinafter, `DTF`).
- `DTF`. It is meant to apply a transformation function to an expression. These enable users to perform advanced transformations on attributes of any data type.

DSL Definition

The DSL has been defined with two main goals: (i) provide versatility so that a wide range of transformations can be carried out, and (ii) reduce the gap between the definition of the transformations and their effects in the target schemas. The syntax of the grammar is given bellow by means of Extended Backus-Naur form notation (Reilly et al. 2003).

`Syntax` is the entry-point to the language. It is given by an `Expression`, which might be one of the following: `Select`, `Index`, `Rename`, `CreateStruct`, `CreateArray`, `Iterate`, or `Transform`.

```
Syntax ::= Expression
Expression ::= Select|Index|Rename|CreateStruct|CreateArray|Iterate|Transform
```

`Select` is intended to be the syntax for selecting either an attribute in a `struct` or a position in an `array`. For instance, regarding the case study A, `t"customerID"` selects the attribute `customerID`, and `t"consumption.[0]"`, selects the position `0` of the `consumption` array.

```
Select ::= 't' '"' ( StringLiteral | '[' Digit+ ']' )
           ( '.' ( StringLiteral | '[' Digit+ ']' ) )* '"'
```

`Rename` is meant to modify the name of an attribute. For example, `"ID" << t"customerID"` changes the name of the attribute `customerID` to `ID`.

```
Rename ::= StringLiteral '<<' Expression
```

`CreateStruct` and `CreateArray` are intended to create a `struct` or an, respectively. For example, `struct("ID" << t"customerID")` creates a `struct` composed of an attribute, `ID`, with the value of the `customerID` attribute. On the other hand, `array(t"customerID")` creates an `array`, with just one position, which contains the value of the `customerID` attribute.

```
CreateStruct ::= 'struct' ' (' Expression ( ',' Expression )* ')'
CreateArray ::= 'array' '(' Expression ( ',' Expression )* ')'
```

`Iterate` enables to perform an operation over an `array` attribute. For instance, `t"consumption" iterate t"startDate"` creates an `array` whose elements are `string` resulting from iterating over the `consumption` attribute and selecting the attribute `startDate`.

```
Iterate ::= Expression 'iterate' Expression
```

`Transform` enables to apply a transformation function to an expression. We have defined nine transformation functions which fit our case study A, but more functions might be added. Regarding the ones defined in `DTF`, they can be classified in two groups: (i) `max`, `min`, `avg`, `sum`, and `subtract`, which are intended to return the maximum, minimum, the average, sum or the subtraction of the values of an attribute, `array` respectively; (ii) `toInt`, `toDouble`, `toString`, and `toDate`, which are intended to cast an attribute to `integer`, `double`, `string`, or `date`, respectively.

```
Transform ::= Expression '->' DTF
DTF ::= 'max'|'min'|'avg'|'sum'|'subtract'|'toInt'|'toDouble'|'toString'|
        'toDate(' StringLiteral')'
```

Transformationss for the Case Study A

Next, the transformations shown in the case study A are performed by means of the DSL we have proposed.

- **T1**. `customerID` is renamed to `ID` by using the `'<<'` operator.

```
"ID" << t"customerID"
```

- **T2**. The matrix `C` is created by iterating over each position in the `consumption array`.

```
"C" << ( t"consumption" iterate array(
           array(t"power.period1", t"power.period4") -> max,
           array(t"power.period2", t"power.period5") -> max,
           array(t"power.period3", t"power.period6") -> max,
       ))
```

- **T3**. Each attribute of `AVG_C` is created by means of the `C` attribute previously created. Each `iterate` operation will result in an `array` with all the values in one of the columns. Then, the average value of each `array` is calculated.

```
"AVG_C" << struct(
            "p1" << (t"C" iterate t"[0]") -> avg,
            "p2" << (t"C" iterate t"[1]") -> avg,
            "p3" << (t"C" iterate t"[2]") -> avg,
           ))
```

- **T4**. `DATES` is created by employing the operator `iterate` and by applying the `toDate` transformation function over the `t"startDate"` attribute.

```
"DATES" << t"consumption" iterate (t"startDate" -> toDate("mm/dd/yyyy"))
```

Transformations for the Case Study B

The transformations for the case study B performed by means of CHAMELEON are as follows.

- **T1** and **T2**. The attributes `accode` and `workstation` are included in the target dataset.

```
                    "accode" << t"aircraft"
                     "workstation" << t"ws"
```

- **T3**. `avg_incidents` is created by iterating over each structure of it. Then, we subtract the resolution date and the start date. The date is previously transformed by using the `toDate` function. Finally, the average of all the subtractions is calculated.

```
"avg_incidents" << (t"incidents" iterate substract(
          t"resolution_date" -> toDate("MM/dd/yyyy HH:mm:ss"),
          t"start_date" -> toDate("MM/dd/yyyy HH:mm:ss")
)) -> avg
```

Benchmark

A set of tests has been devised to evaluate and check the performance of the framework that we have implemented in a Big Data environment. First, the Big Data architecture used to perform the tests is presented. Afterward, we describe the evaluation design to test the performance of our proposal. Finally, the results are drawn and discussed.

Architecture and Implementation

The architecture employed to perform the benchmark is based on a cluster managed by Mesosphere DC/OS (hereinafter DC/OS). DC/OS is an operating system based on Apache Mesos, which enables the execution of technologies for simultaneous data processing. In this case, an Apache Spark cluster has been deployed together with Spark History Server, that enables to extract execution metrics of the Apache Spark applications.

Regarding the infrastructure, it consists of a DC/OS `master` node, responsible for managing the cluster resources and assign them to services, and nine `agents`, responsible for managing the services. The instance of Spark includes a `driver` and nine `executors`. The architecture also includes a node with HDFS to store the datasets and a MongoDB database for storing the execution results. Regarding the computational characteristics, the cluster can reach fifty-two cores between 2 and 2,6 GHz for each and 136 gigabytes of RAM in global. Figure 7 depicts the infrastructure as well as the computational characteristics of the cluster. Summing up, the cluster can reach fifty-two cores and 136 gigabytes of principal memory in global.

Evaluation Design

We have selected the case study A to perform the benchmark, since its schema and transformations are more complex than those of the case study B, so the results will be more reliable. This dataset is composed of approximately more than five million tuples and a size of 2,1 GB. In order to test the scalability of the proposal, nine additional datasets have been created based on two different criteria: (A) four new datasets by increasing the number of tuples; and (B) five new datasets by increasing the size of each tuple (i.e., by increasing the size of the columns), for instance, by duplicating the number of elements in the `consumption` array attribute. Table 1 summarizes the datasets which have been synthetically created by using these two criteria.

Ten test cases have been defined, each of them being executed one hundred times. These tests cases have been classified into two groups of benchmarks: (i) Benchmark 1, where these test cases are intended to check the performance when the dataset size increases by the criteria A; and (ii) the Benchmark 2, where these test cases are intended to check the performance when the dataset size increases by the criteria B. In each test case, all transformations described in the case study A have been applied for each tuple of the dataset.

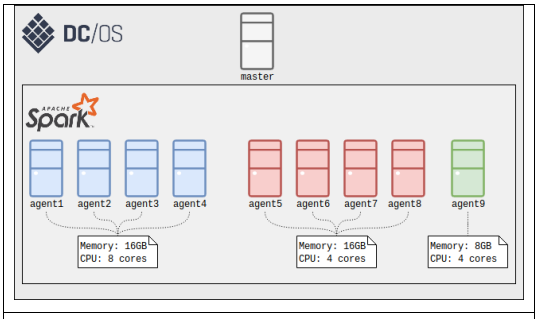


Figure 7. Architecture of the cluster

Table 1. Evaluation Design			
Dataset ID	Criteria	Size (MB)	Benchmark
D1	A	4,119.4	1
D2	A	6,178.4	1
D3	A	8,239.9	1
D4	A	10,299.9	1
D5	B	3,659.8	2
D6	B	5,258.9	2
D7	B	6,859.4	2
D8	B	8,459.2	2
D9	B	10,060.3	2

Table 1. Dataset and Benchmarks for the evaluation

Both benchmarks have been developed by using an Apache Spark application. The application consists of two main stages. The first reads the dataset from HDFS and infers its schema, and the latter distributes the tuples across the cluster, applies the transformations and finally stores the results in MongoDB. As for performance metrics, both the Elapsed Real Time `ERT` and the `CPU Time` of the second stage have been measured in each test case. The `ERT` is the execution time since the stage corresponding to the application of the transformations is launched until it ends. On the other hand, the `CPU Time` is a time accumulator that includes the time the tasks related to the transformations spent on the CPU. For each test case, the average value of one hundred executions will be considered.

Evaluation Results

The results for both benchmarks have been depicted in Figure 8. A trend line has been included in the charts in order to highlight the tendency of the results. The least-squares fitting method has been employed to calculate the trend lines.

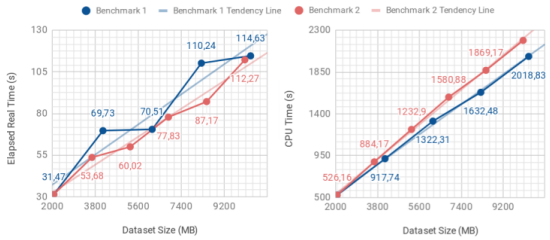


Figure 8. Comparison between Elapsed Real Time (left) and CPU Time (right)

The chart on the left side shows the `ERT` comparison between both benchmarks. The `ERT` tends to be higher in the case of the Benchmark 1. It means that for datasets with the same size, the `ERT` is greater for datasets with more tuples to process than for datasets with less but more complex tuples. This is because the distribution cost is higher when processing a higher number of tuples.

The right-side chart shows the `CPU Time` comparison between both benchmarks. Unlike in the case of the `ERT`, the trend line of the Benchmark 2 tends to be higher than the Benchmark 1.

The trend line equations are (1) for Benchmark 1 whilst for Benchmark 2 it is (2). In fact, the complexity of the dataset used for the Benchmark 2 is higher as its tuples are larger than in Benchmark 1, consuming each one more CPU time. Despite this fact, there is only a 15% of the difference between the slope of the trend lines, being both lines under linear.

$$y = 0.18x + 174 \quad (1) \qquad y = 0.21x + 122 \quad (2)$$

As a conclusion, it is possible to confirm that the proposal scales regarding the dataset complexity because the increment on the complexity on processing nested structures and hence the transformations to apply, only suppose a 15% regarding processing a dataset with less-complex structures.

Related Work

The transformation and querying of complex data have been studied in the fields of database, data warehousing and Big Data. This section describes the state-of-the-art of the transformation of complex data in the ETL and data wrangling fields.

Traditional Approach

Traditionally, the transformation and combination of data sources have been faced up by means of ETL techniques (Arputhamary and Arockiam 2015) in the databases and data warehousing fields. Then, there exist query languages which enable the extraction of complex data. Nevertheless, these transformations are normally meant to give support to a query, and hence, the target schema is not the focus, harvesting the task of defining a specific schema due to the kind of syntax which are typically employed in query languages. The objective of querying is to answer a question on a dataset, and therefore, they usually do not support operators for renaming, restructuring, or transforming complex structures into another complex structure.

Following, four popular query languages with complex data support (Ong et al. 2014) are analyzed. We assess the versatility of the languages, i.e., the operation and transformation capabilities with nested attributes, and the degree in which the query language is aligned with the target schema (i.e., the degree in which the syntax eases the possibility of linking the operations to their effects on performances in relation to the data structure). Both criteria are focused on evaluating the suitability of these languages in a self-service data preparation context.

- **MongoDB**. It is a well-known NoSQL database specialized in semi-structured data. Since its data model enables to deal with complex data, its query language (Botoeva et al. 2018) also supports querying such data, providing operators and function to access nested attributes and operate between them. However, this query language has significant limitations in the definition of custom nested structures, and hence, this language is not aligned with the target schema.
- **N1QL**. It is the query language of Couchbase (Ostrovsky et al. 2015), a NoSQL database. This is a SQL-like language, adding operators and functions to operate with columns with nested structures. Due to its orientation toward SQL, the syntax of the language is not aligned with the definition of the target schema.
- **JSONiq**. It is a JSON-oriented scripting query language (Florescu and Fourny 2013), based on a JSON-like data model. Although this language offers great versatility, the definition of the queries is not aligned with the target schema and the input data type is restricted to JSON.
- **Jaql**. This query language (Beyer et al. 2011) is intended to query semi-structured data in Hadoop. As JSONiq, it is also a scripting language, offering good versatility and hence supporting operators to perform a wide range of transformations. In addition, the language enables to align its queries to the target schema.

While the most versatile languages are JSONiq and Jaql, the one with the best syntax for our purpose is Jaql. However, we find areas of improvement in relation to the complexity and the syntax. First, the great versatility of this language leads to a complexity that, in our opinion, can be decreased. Second, although the flexibility of the syntax enables users to align the transformations with the effects in the target schema, we think that it is possible to achieve a better alignment between them.

Data Wrangling

In recent years, Academia and Industry have used the term data wrangling to refer to the transformation, combination and cleaning of data in an exploratory way (Furche et al. 2016). The current trend is to make this task easier so that it can be carried out by non-expert users. Consequently, the latest proposals that can be found in the literature are focused on assisting users in this process (a.k.a. self-service), offering features such as (i) data profiling, so that the user can explore the data and obtain a holistic view of them (e.g., see the quality of them at first glance); (ii) suggest transformations based on a knowledge base, or based on criteria that may improve the quality of the data; and (iii) automatically infer transformations by means of example process results.

Although there exist data wrangling solutions in Industry, just a few of them support obtaining data from semi-structured data sources, transforming them, and exporting the dataset with complex data structures. In the study we have carried out, we have considered two of the most influential tools in academia and in the industry that enable to deal with complex data: Trifacta (a.k.a. Google Cloud Dataprep) and OpenRefine.

- **Trifacta**. It was originated in Academia, conceived as a visual data wrangling tool with a transformation language known as data-wrangler (Kandel et al. 2011). Now, it is a commercial tool, and one of the references in the self-service data preparation field. In relation to the transformation of complex data, it supports nesting and unnesting operators.
- **OpenRefine**. It was originally maintained by Google (Kusumasari and Fitria 2016). Now, it is an open source tool that has been employed in multiple research works. It provides a query language known as GREL (Google Refine Expression Language). It is a Java Script-based language which enables the transformation of complex data.

First, the Trifacta data model is table-oriented. For this reason, transformation operations are carried out on columns. This implies that in data with nested structures, only those that are in the top-level can be operated. This has two major drawbacks: (i) Data profiling functionalities, data quality analysis, and transformation suggestions do not reach those attributes that are nested; and (ii) in order to perform any type of operation between attributes that are within the same nested structure, it is required to perform as many unnesting operations as how deep is the attribute to be employed. This inevitably increases the complexity of the transformations and the ease of making mistakes. OpenRefine poses similar problems. In this case, users must employ a Java Script-based language to perform the operations in a programmatic way. Although it offers a good versatility, it does not fulfill the criteria we defined above, being far from the

objectives of the self-service data preparation. In addition to these drawbacks, there is a lack of support for data profiling, data quality, and transformation suggestions and inferences based on complex data.

To the best of our knowledge, our proposal is the first attempt to contribute to the inclusion of semi-structured complex data in the data wrangling and self-service data preparation fields.

Comparison with Data Wrangling Tools

The objective of this section is to evidence the drawbacks of the current data wrangling tools when dealing with complex data. We also point at how our proposal could improve the transformation of complex data from the point of view of the user.

In the Related Work section, we considered two of the most influential data wrangling tools in the Industry and Academy (Trifacta and OpenRefine). However, Trifacta includes and improves the functionalities offered by the other proposals. Therefore, the analysis of Trifacta implies the analysis of the functionalities of their competitors, since Trifacta is the more complete tool in data wrangling context. On the one hand, Trifacta's data model is table-oriented, being unable to represent data with more than one dimensionality (i.e., nested structures). Despite this, it is possible to transform nested attributes as well as nesting other attributes, but it will require the user to flatten arrays, or unnest attributes by creating new columns. On the other hand, OpenRefine is also a table-oriented solution, but it is unable to nest attributes, making it impossible to create columns of data type struct. To work around it, OpenRefine allows users to employ an imperative Java Script-based language, as explained in the Related Work section. Since it is an imperative language, it is out of our scope. For this reason, and since it is one of the most complete data wrangling tools in Industry, we have selected Trifacta to show how our proposal could improve the transformation of complex data.

In order to show the comparison, we have resolved the cases studies presented in this paper by using Trifacta. Then, for each case study we have created a table which depicts the number of operations which are necessary to complete each single transformation, classified by type. The types of operations that we have considered are: (i) **unnesting**, which consists of extracting nested attributes in columns of type struct, resulting in an augmentation in the number of columns; (ii) **nesting**, which joins several columns in a single struct column; (iii) **flattening**, which is applied to columns of type array, resulting in an augmentation in the number of rows; (iv) **grouping**, which reduces the number of rows by grouping them according to a criteria; (v) **dropping**, which deletes a set of columns specified by the user; (vi) **renaming**, that includes those renaming operations that must be performed due to column name changes that might occur during intermediate operations, and (vii) **non-intermediate**, that includes simple operations such as column renaming, data formatting, arithmetic operations and operations with lists of numbers.

The most important aspect here is that in order to access nested attributes in a data schema, it is required to isolate those attributes in single columns, and once they have been transformed, they must be sent back to their corresponding nested structure. We consider that these intermediate operations complicate the transformation process of complex data. To illustrate this problem, Figure 9 represents the sequence of operations that are required in order to carry out the transformation T2 of case study A with Trifacta. As can be seen, in order to access to the attributes inside `consumption.power`, two intermediate operations are required (flattening and unnesting). Then, three additional operations (three nesting) are required before calculating the maximum value for each period, and finally, in order to create the `C` matrix, two additional intermediate operations are required (nesting and grouping). In addition to these intermediate operations, three operations (two dropping and one renaming) must be performed in order to deal with temporary columns and column names that are generated during the transformation process. In total, T2 requires 13 operations, where 10 of them are intermediate operations, which means that the 77% of the operations that the user must perform to carry out this transformation are intermediate operations needed to access the attributes and to give them the right structure. The objective of our proposal is to abstract users from these intermediate operations so that they just have to navigate through the structure in order to operate with the desired attributes, and at the same time they can change the structure of the dataset. We believe that, in this way, the transformations are better aligned with their results in the target schema, making the transformation process more intuitive and in concordance with the objectives of data wrangling.

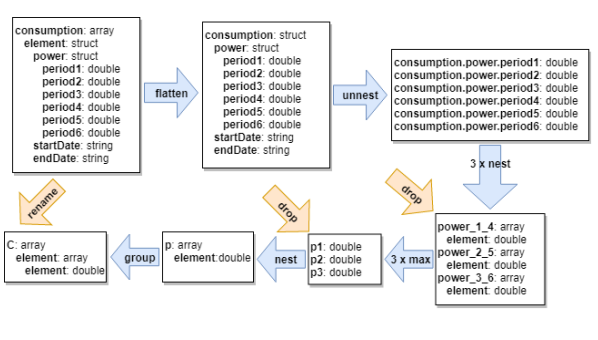


Figure 9. Representation of the intermediate steps needed for the transformation T2 in case study A

Case Study A

Table 2 shows the operations performed to complete the case study A with Trifacta. This case study requires 34 operations, where 26 of them are intermediate operations and 8 of them are not. Approximately the 77% of the operations written by the user are intermediate. Transformations T2 and T3 have a similar proportion of intermediate operations, while in T4 this percentage increases to the 85%. Our proposal abstracts users from these intermediate operations, so we can state that, except for T1 (that does not require to transform nested attributes), it improves the way the transformations are carried out.

Table 2. Operations to complete Case Study A with Trifacta

	Unnesting	Nesting	Flattening	Grouping	Dropping	Renaming	Non-intermediate	Total
T1	0	0	0	0	0	0	1	1
T2	1	4	1	1	2	1	3	13
T3	1	4	1	1	2	1	3	13
T4	1	1	1	1	1	1	1	7
TOTAL	3	9	3	3	5	3	8	34

Table 2. Operations to complete Case Study A with Trifacta

Case Study B

Table 3 shows the operations performed to complete case study B with Trifacta. The most complex transformation here is T3, which is the only one that requires access to nested attributes. It requires a total of 5 operations, being 4 of them intermediate operations. Hence, the 80% of the operations for T3 are intermediate operations. Ultimately, the 57% of the operations of the case study are intermediate. Our proposal also improves the transformation process for this case study.

Table 3. Operations to complete Case Study B with Trifacta								
	Unnesting	Nesting	Flattening	Grouping	Dropping	Renaming	Non-intermediate	Total
T1	0	0	0	0	0	0	1	1
T2	0	0	0	0	0	0	1	1
T3	1	0	1	1	0	1	1	5
TOTAL	1	0	1	1	0	1	3	7

Table 3. Operations to complete Case Study B with Trifacta

Conclusions

One of the most important challenges that last advances in Industry poses is the transformation of complex data and the conciliation of complex data schemata. There is an emerging field in Big Data which intends to ease these tasks for non-expert users: data wrangling and self-service data preparation. This study has highlighted the lack of support for complex data in these fields.

Our proposal is intended to contribute to the improvement of data wrangling techniques by means of a framework that includes a Domain-Specific Language. The goal of it is to link the operations carried out by users to their effects in the target schema. It represents an improvement in relation to the data wrangling solutions that can be found in Industry since they are not focused on dealing with complex data. Several well-known query languages for semi-structured data have been studied to enhance our proposal.

Future Work

The limitations of our proposal define the actions that we want to face up for the future. First, our proposal lacks support for data profiling, data quality assessment, and automatic assistance to users. These shortcomings are hence a great opportunity for the future, since as proven in this study, data wrangling and self-service data preparation are of paramount relevance in both academia and Industry.

In particular, we identify prior opportunities (Furche et al. 2016) in the automation of error-detection and feedback giving in the definition of transformation rules by users. It would be a pioneering proposal in the field of semi-structured complex data.

Acknowledgments

This work has been partially funded by the Ministry of Science and Technology of Spain by ECLIPSE project (RTI2018-094283-B-C33) and the European Regional Development Fund (ERDF/FEDER) via METAMORFOSIS project.

References

Ardagna, C. A., Bellandi, V., Bezzi, M., Ceravolo, P., Damiani, E., and Hebert, C. 2018. "Model-Based Big Data Analytics-as-a-Service: Take Big Data to the Next Level," *IEEE Transactions on Services Computing*, pp. 1–1. (https://doi.org/10.1109/TSC.2018.2816941).

Ardagna, C. A., Bellandi, V., Ceravolo, P., Damiani, E., Di Martino, B., D'Angelo, S., and Esposito, A. 2018. "A Fast and Incremental Development Life Cycle for Data Analytics as a Service," in *2018 IEEE International Congress on Big Data (BigData Congress)*, IEEE, July, pp. 174–181. (https://doi.org/10.1109/BigDataCongress.2018.00030).

Arputhamary, B., and Arockiam, L. 2015. "Data Integration in Big Data Environment," *Bonfring International Journal of Data Mining* (5:1), Bonfring, pp. 01–05. (https://doi.org/10.9756/BIJDM.8001).

Beyer, K. S., Ercegovac, V., Gemulla, R., Balmin, A., Eltabakh, M. Y., Kanne, C.-C., Özcan, F., and Shekita,

E. J. 2011. "Jaql: A Scripting Language for Large Scale Semistructured Data Analysis," in *Proceedings of VLDB Conference*. (https://www.semanticscholar.org/paper/Jaql%3A-A-Scripting-Language-for-Large-Scale-Data-Beyer-Ercegovac/28d0280e2b972155c203b96bd6eb9f826aa73850).

Botoeva, E., Calvanese, D., Cogrel, B., and Xiao, G. 2018. "Expressivity and Complexity of MongoDB Queries," *DROPS-IDN/8607*, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik. (https://doi.org/10.4230/LIPICS.ICDT.2018.9).

Ceravolo, P., Azzini, A., Angelini, M., Catarci, T., Cudré-Mauroux, P., Damiani, E., Mazak, A., Van Keulen, M., Jarrar, M., Santucci, G., Sattler, K. U., Scannapieco, M., Wimmer, M., Wrembel, R., and Zaraket, F. 2018. "Big Data Semantics," *Journal on Data Semantics* (7:2), Springer Berlin Heidelberg, pp. 65–85. (https://doi.org/10.1007/s13740-018-0086-2).

Dong, X. L., and Srivastava, D. 2015. "Big Data Integration," *Synthesis Lectures on Data Management* (7:1), Morgan & Claypool Publishers, pp. 1–198. (https://doi.org/10.2200/S00578ED1V01Y201404DTM040).

Florescu, D., and Fourny, G. 2013. "JSONiq: The History of a Query Language," *IEEE Internet Computing* (17:5), pp. 86–90. (https://doi.org/10.1109/MIC.2013.97).

Furche, T., Gottlob, G., Libkin, L., Orsi, G., and Paton, N. W. 2016. "Data Wrangling for Big Data: Challenges and Opportunities," in *19th International Conference on Extending Database Technology (EDBT)*, pp. 473–478. (https://doi.org/10.5441/002/EDBT.2016.44).

Gerbert, P., Lorenz, M., Rüßmann, M., Waldner, M., Justus, J., Engel, P., and Harnisch, M. 2015. "Industry 4.0: The Future of Productivity and Growth in Manufacturing Industries," *Boston Consulting Group*. (https://www.bcg.com/publications/2015/engineered_products_project_business_industry_4_future_productivity_growth_manufacturing_industries.aspx, accessed December 19, 2018).

Guo, P. J., Kandel, S., Hellerstein, J. M., and Heer, J. 2011. "Proactive Wrangling: Mixed-Initiative End-User Programming of Data Transformation Scripts," in *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology - UIST '11*, New York, New York, USA: ACM Press, p. 65. (https://doi.org/10.1145/2047196.2047205).

Hellerstein, J. M., Heer, J., and Kandel, S. 2018. "Self-Service Data Preparation: Research to Practice," *Undefined*. (https://www.semanticscholar.org/paper/Self-Service-Data-Preparation%3A-Research-to-Practice-Hellerstein-Heer/715cba311d4e5ad6b5f8cba7694ccc03ef7583b7).

Jin, Z., Anderson, M. R., Cafarella, M., and Jagadish, H. V. 2017. "Foofah: Transforming Data By Example," in *Proceedings of the 2017 ACM International Conference on Management of Data - SIGMOD '17*, New York, New York, USA: ACM Press, pp. 683–698. (https://doi.org/10.1145/3035918.3064034).

Kandel, S., Paepcke, A., Hellerstein, J., and Heer, J. 2011. "Wrangler: Interactive Visual Specification of Data Transformation Scripts," in *Proceedings of the 2011 Annual Conference on Human Factors in Computing Systems - CHI '11*, New York, New York, USA: ACM Press, p. 3363. (https://doi.org/10.1145/1978942.1979444).

Kusumasari, T. F., and Fitria. 2016. "Data Profiling for Data Quality Improvement with OpenRefine," in *2016 International Conference on Information Technology Systems and Innovation (ICITSI)*, IEEE, October, pp. 1–6. (https://doi.org/10.1109/ICITSI.2016.7858197).

Lee, I. 2017. "Big Data: Dimensions, Evolution, Impacts, and Challenges," *Business Horizons* (60:3), Elsevier, pp. 293–303. (https://doi.org/10.1016/J.BUSHOR.2017.01.004).

Milutinovic, V., Kotlar, M., Stojanovic, M., Dundic, I., Trifunovic, N., and Babovic, Z. 2017. *DataFlow Systems: From Their Origins to Future Applications in Data Analytics, Deep Learning, and the Internet of Things*, Springer, Cham, pp. 127–148. (https://doi.org/10.1007/978-3-319-66125-4_5).

Obitko, M., and Jirkovský, V. 2015. *Big Data Semantics in Industry 4.0*, Springer, Cham, pp. 217–229. (https://doi.org/10.1007/978-3-319-22867-9_19).

Ong, K. W., Papakonstantinou, Y., and Vernoux, R. 2014. "The SQL++ Semi-Structured Data Model and Query Language: A Capabilities Survey of Sql-on-Hadoop, Nosql and Newsql Databases," *CoRR, Abs/1405.3631*.

Ostrovsky, D., Haji, M., and Rodenski, Y. 2015. "The N1QL Query Language," in *Pro Couchbase Server*, Berkeley, CA: Apress, pp. 107–133. (https://doi.org/10.1007/978-1-4842-1185-4_6).

Parody, L., Vaca, Á. V., Gómez-López, M., and Gasca, R. 2017. "FABIOLA: Defining the Components for Constraint Optimization Problems in Big Data Environment," in *International Conference on Information Systems Development (ISD) 2017*, , September 26. (https://aisel.aisnet.org/isd2014/proceedings2017/CogScience/3).

Reilly, E. D., Ralston, A., and Hemmendinger, D. 2003. "Backus-Naur Form (BNF)," in *Encyclopedia of Computer Science*, Wiley, pp. 129–131. (https://dl.acm.org/citation.cfm?id=1074155).

Riehle, D., Dirk, Riehle, and Dirk. 1997. "Composite Design Patterns," *ACM SIGPLAN Notices* (32:10), ACM, pp. 218–228. (https://doi.org/10.1145/263700.263739).

Stefanowski, J., Krawiec, K., and Wrembel, R. 2017. "Exploring Complex and Big Data," *International Journal of Applied Mathematics and Computer Science* (27:4), Walter de Gruyter & Co., pp. 669–679. (https://doi.org/10.1515/amcs-2017-0046).

Valencia-Parra, Á., Ramos-Gutiérrez, B., Varela-Vaca, Á. J., Gómez-López, M. T., and Gracía Bernal, A. 2019. "Enabling Process Mining in Aircraft Manufactures: Extracting Event Logs and Discovering Processes from Complex Data," in *Proceedings of the Industry Forum at BPM 2019*, Vienna, pp. 166–177.

3.2.2 DMN4DQ: When data quality meets DMN

Published in the Decision Support Systems journal (Vol. 141, p.113450). Elsevier BV.

- ***Authors**: Álvaro Valencia-Parra, Luisa Parody, Ángel Jesús Verala-Vaca, Ismael Caballero, María Teresa Gómez-López.*
- ***DOI**: 10.1016/j.dss.2020.113450.*
- ***Rating**: Q1 (JCR'20 5.795).*

Decision Support Systems 141 (2021) 113450

Contents lists available at ScienceDirect

Decision Support Systems

journal homepage: www.elsevier.com/locate/dss

DMN4DQ: When data quality meets DMN

Álvaro Valencia-Parra [a,*], Luisa Parody [b], Ángel Jesús Varela-Vaca [a], Ismael Caballero [c], María Teresa Gómez-López [a]

[a] Dpto. Lenguajes y Sistemas Informáticos, Universidad de Sevilla, Sevilla, Spain
[b] Dpto. Métodos Cuantitativos, Universidad Loyola Andalucía, Sevilla, Spain
[c] Information Systems and Technologies Institute, University of Castilla-La Mancha, Ciudad Real, Spain

ARTICLE INFO

Keywords:
Data usability
Data quality
Decision model and notation
Data quality rule
Data quality assessment
Data quality measurement

ABSTRACT

To succeed in their business processes, organizations need data that not only attains suitable levels of quality for the task at hand, but that can also be considered as usable for the business. However, many researchers ground the potential usability of the data on its quality. Organizations would benefit from receiving recommendations on the usability of the data before its use. We propose that the recommendation on the usability of the data be supported by a decision process, which includes a context-dependent data-quality assessment based on business rules. Ideally, this recommendation would be generated automatically. Decision Model and Notation (DMN) enables the assessment of data quality based on the evaluation of business rules, and also, provides stakeholders (e.g., data stewards) with sound support for the automation of the whole process of generation of a recommendation regarding usability based on data quality.

The main contribution of the proposal involves designing and enabling both DMN-driven mechanisms and a guiding methodology (DMN4DQ) to support the automatic generation of a decision-based recommendation on the potential usability of a data record in terms of its level of data quality. Furthermore, the validation of the proposal is performed through the application of a real dataset.

1. Introduction

The witnessed changes that Digital Transformation (e.g., Industry 4.0) is introducing in different business processes across various domains have positioned data at the core of operations and strategies [33]. To a certain extent, it can be stated that the role previously played by steam engines in Industry 1.0 is now played by the new and powerful AI-based machines [12]. However, and as happened in those times, the success of these new AI-machines, and therefore, of business processes, largely relies on the quality of the raw material employed, in this case, data. Consequently, the management of the quality of data has become essential in this digital era [20,25].

Given the need for data with adequate levels of quality in such domains, we propose that if organizations could automatically incorporate ways to decide on whether to use or discard records, then business processes would greatly benefit from preventing results that would otherwise produce low levels of data quality. This decision regarding the potential usability of the data could be made after the generation of a recommendation based on the assessment of the quality of the data records.

Since it is generally accepted that the assessment of data quality is context-dependent [3,15], and since we propose that the usability of the data largely depends on the quality of the data, it can therefore be stated that the usability of data is also largely dependent on the context of the use of the data [14,37,44]. This implies modelling the context in which the data is to be used and when a data record is potentially usable.

In order to convert this idea into action, we conducted an investigation to tackle two challenges: (i) how to describe whether a data record is usable for its intended use in a given context; and (ii) how to automate the process of producing a recommendation on the usability of the data for this context.

To deal with the first challenge, we studied how others had already faced the problem of modelling the context, the data, and the rules that describe when a data record is of sufficient quality by identifying and describing various types of business rules for data quality and how the recommendation of the usability of the data could be determined. In

* Corresponding author.
E-mail addresses: avalencia@us.es (Á. Valencia-Parra), mlparody@uloyola.es (L. Parody), ajvarela@us.es (Á.J. Varela-Vaca), Ismael.Caballero@uclm.es (I. Caballero), maytegomez@us.es (M.T. Gómez-López).

https://doi.org/10.1016/j.dss.2020.113450
Received 10 June 2020; Received in revised form 15 November 2020; Accepted 15 November 2020
Available online 18 November 2020

order to ensure rigour and repeatability on the process, we decided to incorporate all the elements identified and the necessary steps into a methodology.

The second challenge to address is the necessity to support the generation of the recommendation on the usability of the data, based on its levels of quality in an automatic and technological-agnostic way [35]. This is even more challenging in scenarios where high efficiency is required in terms of computational cost, such as in Internet of Things (IoT) or in the context of CyberPhysical Systems (CPS). As part of this second challenge, and to set our proposal in motion, we suggest the use of a solution that facilitates the description and validation of the business rules employed in the assessment of the level of usability based on data quality, thereby promoting the application of repeatable decisions that can be semantically interoperable with the various technologies through which data quality assessment could be applied. We found that in order to tackle these challenges, it was recommendable to use a decision language that facilitates the description of the business rules so that they could be verified automatically. In this respect, OMG's Decision Model and Notation (DMN) [30] and the FEEL expression language for modelling conditions could prove themselves to be perfect allies in achieving these two challenges. For this reason, DMN is the main pillar of the structure of proposal.

Therefore, the main contribution of the proposal involves *designing and enabling DMN-driven mechanisms to support the automatic generation of a business-based recommendation on the potential usability of a data record in terms of its level of data quality*. To this end, our proposal includes the following actions:

1. Development of the foundations of the proposal through a set of integrated and hierarchical business rules that address the concepts of data quality measurement and data quality assessment for the generation of a data-usability recommendation (Section 3.1).
2. Identification and tailoring of the necessary elements provided by the standard DMN to support our proposal (Section 3.2).
3. Definition of a methodology, called DMN4DQ, to enable data-related users (e.g., stakeholders and data stewards) to drive the process of instantiating the corresponding elements when it comes to producing recommendations for a given dataset in a given context. This includes the definition and implementation of a software architecture supported by commercial implementations of reference (e.g., Camunda DMN) to automate the process of generating the recommendation (Section 4).
4. Validation of the proposal in a case study with real data (Section 5).

The remainder of the paper is organised as follows: Section 6 shows related work; Section 7 analyses threats to the validity of the approach; and finally, Section 8 presents concluding remarks and lessons learned.

2. Foundations

In order to combine DMN and data quality, and before detailing our proposal, it is necessary to revisit certain concepts regarding data quality management and DMN to enable a better understanding of how DMN can be used to describe whether a data record is usable in terms of its level of quality in a given context.

2.1. Data quality management: measurement and assessment

Throughout the literature, the two most widely used definitions of *"data quality"* are based on the notions of *"meeting requirements"* (i.e., a measure of the numbers of defects) given by Crosby, and *"fitness for use"* coined by Juran [38]. From our understanding, these two definitions involve a major difference: while the first definition enables somebody to *measure "how well data is built"* (for instance, by counting the number of times that the data fails to meet stated requirements), the second lets somebody *assess "how usable the data is"* in a given context by comparing the number of defects found (the "measures") with a threshold value representing the appetite for risk of the organisation regarding the reliability of the data in an specific context [8].

Even though the terms "measurement" and "assessment" can sometimes be considered as synonyms, we highlight this difference because it is important for our proposal: the "assessment" requires the "measurement", in the same way that the "generation of a recommendation on the usability" requires the "assessment". Our proposal goes a step beyond, since before determining whether a record is potentially usable, it is necessary to make the most important decision: *While taking into account the impact that using data with inadequate levels of quality can have on the success of the business processes, should the assessed data record be used or discarded in the context of the task at hand?*. If the use of the data is potentially risky for the business, then data stewards may decide: to enhance the data (e.g., data cleansing); to use the unaltered data, thereby assuming a risk; or alternatively, to discard the data record. The main aim of our proposal is therefore to provide business-based recommendations to data stewards to facilitate decision-making on whether to use or discard the data as part of their business activities. Therefore, there is a patent need to manage data quality. The concept of data quality dimension (also called data quality characteristic) lies at the core of data quality management. A data quality dimension can be understood as *a criterion employed to evaluate the quality of data* [28,37,44]. These dimensions or characteristics represent the data quality requirements stated or expected by the various stakeholders involved in the execution of the business processes [45]. A set of data quality dimensions is called a data quality model. Several researchers and practitioners in a variety of contexts have proposed their own data quality models [32,36]. Due to its importance at different stages of the data quality management discipline, we would like to highlight two generic models from among all the existing models: (1) the model proposed by Wang et al. [43] (see Table 1); and (2) the model proposed in ISO 25012 [21] (see Table 2).

The first model has been the most widely used in recent years since it is the most authoritative reference in the field. Moreover, it guides the identification of the specific data quality requirements that are important for a given context. In order to validate the compliance of these requirements, business rules are typically employed in data quality contexts [7,10,34].

The second model, ISO 25012, should not necessarily be understood as an alternative to the proposal of Wang et al. In fact, in conjunction with ISO 25024 [22], it complements their model by providing important indications for the definition of measurements and measurement methods for the data quality dimensions or characteristics. ISO 25012 introduces fifteen data quality characteristics, which are classified into the following three groups: (i) Inherent. The definition for these data quality characteristics is introduced in Table 2; (ii) System dependent. There are various characteristics whose measurement or assessment largely depends on the implementation of the systems in which data is stored, retrieved or processed; (iii) Inherent and system dependent. This group contains some of the previous data quality characteristics whose measurement and/or assessment can be subject to a two-fold interpretation based on the ideas introduced in the two previous groups.

Please, note that Wang et al. (and many other investigations based on this seminal work) use the term "dimension", whereas in the standards,

Table 1
Data quality dimensions by Wang et al. [43].

Data quality category	Data quality dimension
Intrinsic	Accuracy, Objectivity, Believability, Reputation
Accessibility	Access, Security
Contextual	Relevancy, Value-Added, Timeliness, Completeness, Amount of data
Representational	Interpretability, Ease of understanding, Concise representation, and Consistent representation

Table 2
Definition of the inherent data quality characteristics from ISO 25012 [21].

Data quality characteristic	Definition
Accuracy	The degree to which data has attributes that correctly represent the true value of the intended attribute of a concept or event in a specific context of use.
Completeness	The degree to which subject data associated with an entity has values for all expected attributes and related entity instances in a specific context of use.
Consistency	The degree to which data has attributes that are free from contradiction and are coherent with other data in a specific context of use.
Credibility	The degree to which data has attributes that are regarded as true and believable by users in a specific context of use.
Currentness	The degree to which data has attributes that are of the right age in a specific context of use.

the term "characteristic" is preferred. Even though it would be possible to justify the difference, for the sake of the simplicity, let us consider these two terms as synonymous in this manuscript.

As previously stated, measurement and assessment of data quality characteristics is a complex process, which largely depends on the context of use of the data. The context of the data includes: the organisational environment and the description of the business processes in which data is used; the technological architecture supporting the use of the data; and the skills and knowledge of both data stewards and data quality analysts in charge of managing or using the data [8,24]. In order to deal with this complexity, several researchers and practitioners have proposed different methodologies, which include a variety of metrics and/or measurement methods [4,29,34,43].

Typically, measurement methods that are to be applied first require relational datasets to be profiled for a better understanding of the nature of the data. These profiling processes are often performed in batch activities that include all data records in the dataset and by means of data profiling tools [13,34]. Nevertheless, in order to measure data quality, most authors have developed their own rule-based data quality measurement systems. The foundations of these rule-based data quality measurement systems have been formalised by Bronselaer et al. in [7]. The creation of rule-based data quality measurement systems involves that stakeholders (i.e. data stewards) should conform to the semantics of the data and their context. However, if stakeholders remain unaware of the semantics of the data and the way in which these semantics have been implemented throughout the various data models (conceptual, logical, and physical), then the application of this kind of tool will require an extra effort towards diagnosing the root causes of the low levels of data quality, since the methodology behind these tools fails to contemplate the context in which the data quality process is applied. The instantiating of these frameworks involves the creation of various elements, such as the set of business rules to which data must adhere, and the possible results of the measurement that the capacity function should produce. An example of the instantiating of this conceptualization for the measurement of the level of quality in information systems research is presented in [39], wherein Timmerman and Bronselaer, after reviewing the foundations of rule-based data quality measurement, present a rule-based framework for the measurement and assessment of the quality of data in Information Systems research. As we will explain in Section 3, our proposal, goes beyond the measurement of data quality, involving more stages. Consequently, we have to describe and relate several sets of business rules.

2.2. Decision model and notation

Decision Model and Notation (DMN) is the modelling language and notation standard defined by OMG to describe decision rules [30]. Thus, DMN is a standard approach that facilitates the modelling of repeatable decisions. The decisions can be customised according to the necessities of each organisation or moment, thereby ensuring that the decision models are interchangeable. DMN facilitates the declarative description and formalisation of the decisions with the form "if-then" [19]. Furthermore, since DMN is supported by a set of engines, such as Camunda - DMN Engine[1] and Drools - DMN Engine,[2] we found DMN to be the most suitable concept to come to the data quality assessment to real applicability of it.

DMN provides a mechanism to define a decision logic model that is understandable by non-expert users (i.e., business data stewards in charge of describing the data quality requirements). In addition, DMN enables the separation of the decision logic from the control-flow logic, thereby centralising the conditional expressions that guide the decisions.

The DMN standard provides two customisable components: the Decision requirement diagram, which enables the definition to be made of the decisions to be taken, of their interrelationships, and of their requirements for decision logic; and the Decision logic, which allows the representation of the required decisions with sufficient details to enable validation and/or automation.

Decision logic is described by means of a *decision table* (see Fig. 1), which includes a set of inputs, decision rules, and output values. In a horizontal representation of the rules (an equivalent vertical representation is also possible), the input and outputs are defined in columns and the rules as rows. Each IF-THEN condition is represented in a row, as a conjunction of basic expressions written in FEEL (Friendly Enough Expression Language) [30]. The output returns the values of the row that is satisfiable according to the input. The example considers three features given as input data (i.e., *CPU*, *Memory*, and *Storage*), which returns a decision as output for the variable *Instance_Family*.

Each condition that appears in the DMN table (such as $>= 2.9$) relies on FEEL,[3] an expression language that enables the writing of the conditions for the rules in the DMN tables. FEEL supports several data types as input and output values (e.g., String, Integer, Decimal, Date, Boolean), and implements a set of built-in functions to write more complex conditions on the input values. In addition, this expression language also supports null values and conditions that are always true ('–'). Users can modify the behaviour of the built-in functions as well as creating their own functions to better adjust the rules to the nature of their data. This versatility ensures that any of the types of business rules defined in Section 3.1 can be modelled with this technology, and formalised as described in Section 3.2.

The *information item name* is the name of the variable for which the decision table provides the decision logic. The *hit policy indicator* determines how to handle the multiple matches of the rules described in DMN [30]. This indicator takes any of 5 values: *Unique (U):* only one rule can be triggered, and it is not possible that more than one can satisfy a

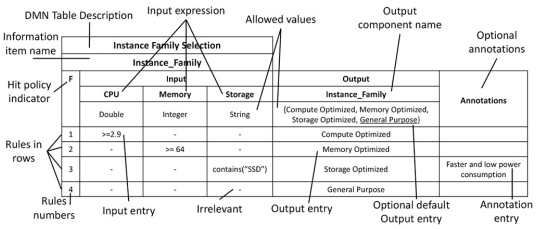


Fig. 1. Decision table for selecting the *Instance Family* depending on the *CPU*, *Memory*, and *Storage* of the server.

[1] https://camunda.com/products/dmn-engine/
[2] https://www.drools.org/learn/dmn.html
[3] https://docs.camunda.org/manual/7.4/reference/dmn11/feel/

rule for an input tuple; *Any (A):* Multiple rules can be triggered, but they must agree in the output; *Priority (P):* Multiple rules can be triggered, and the output corresponds to the rule that has the highest priority; *First (F):* Multiple rules can be triggered, and the output corresponds to the order of the rows in the decision table; and *Collect (C):* Multiple rules can be triggered, and the output is an aggregation.

In our case, *F* means that although multiple rules can match, only the first hit by rule order is returned. Finally, the possible results of this decision table are "*Compute optimised*", "*Memory optimised*", "*Storage optimised*", and "*General purpose*". Table 3 depicts the results of applying the DMN table presented in Fig. 1.

Decision tables can be combined in a hierarchical way, in that the output of one table can be the input of another further up in the hierarchy. For instance, as shown in Fig. 2, the output of "Instance Family Selection" will be used as input of the decision table "Supplier Selection".

Unlike other alternatives for the verification of business rules, DMN allows users to define their rules in a hierarchical way, thereby maintaining the coherence with the logical structure previously explained. The way in which rules are graphically modelled, its versatility, and the possibility of automating the evaluation of the rules, make DMN the ideal candidate for technically supporting our methodology.

3. Rationale of DMN4DQ

Given the goal of our investigation, in this section, the rationale of our proposal is presented. Firstly, we describe the conceptualization of how to generate the recommendation on the usability of a data record based on its level of quality. Subsequently, an explanation is given on how DMN has been employed to represent and evaluate the required sets of business rules to automate the generation of a recommendation for the use of a data record.

3.1. Business rules for the determination of the usability of the data records

The generation of a recommendation of a business-driven decision on using or discarding a data record can be based on several criteria, but, to our understanding, data quality is the most important criterion because decisions are no better than the data on which they are based [44]. Consequently, to generate this recommendation, it is necessary to assess the level of data quality within a context of use. As stated in Section 2.1, this assessment requires a previous stage of measurement of data quality. Both the assessment and measurement require different data quality dimensions that represent stakeholders' data quality requirements for each task at hand.

It is known that business rules define certain business constraints of an organisation, such as who can execute an action, the order of the activities, and the acceptable thresholds for specific KPIs. In this respect, it is possible to define business rules to address data quality concerns for the proper execution of the processes of an organisation. Hence, business rules for data quality gather the knowledge acquired by an organisation to reflect when a data record can provide value for specific business goals. There exists a consensus that business rules are an effective way to control data quality [2], and the term "Data Quality Rule" has been used in the context of data quality management [27,34].

Our proposal introduces the notion of Business Rules for Data Decisions (BR.DD), that must be defined in order to ascertain the usability of a data record. The types of BR.DD considered in our proposal are the following:

- The "Business Rules for Data Values" (BR.DV) are those aimed towards evaluating the extent to which a data requirement is met. An example of BR.DV is provided in the following: Given that the length of a String must be longer than 6, if the length of an input String is from 3 to 6 then it returns 'A', else if it is from 6 to 9 then it returns 'B', and 'C' is returned otherwise. Semantically, 'A' is intended to represent the lowest level of fulfilment, and 'C' represents a suitable level of fulfilment, with 'B' representing an average term. We consider that this is not a proper data quality measurement since data quality dimensions have yet to be involved. However, the output of this evaluation will be the input of the specific data quality measurement.
- The "Business Rules for Data Quality Measurement" (BR.DQM) are those rules employed to compute the measurement of the level of quality of each data quality dimension according to the BR.DV. For example, a BR.DQM for the *accuracy* dimension could be stated as follows: A record can be considered as *Dramatically Non-Accurate* if the output of BR.DV.01 is 'A', and *Accurate* if the output of BR.DV.01 is 'B' or 'C'.
- The "Business Rules for Data Quality Assessment" (BR.DQA) are those rules that describe the assessment of the data quality in accordance with a set of BR.DQM by combining the results of the measurement of several data quality dimensions, as indicated by the business. An example of BR.DQA is A record can be considered as: *Usable but assuming High Risk* if it is *Accurate* or *Correct*; *Usable and assuming Low Risk*, if it is *Accurate* and *Correct*; and *non-usable* otherwise.
- The "Business Rules for Data Usability Decision" (BR.DUD) are those rules employed to generate the recommendation about using or discarding the data record for the intended use based on the assessment of its level of data quality. At this point, the organisational risk-appetite of the organisation should be considered with regard to the use of this specific data record. For instance, A record will be *used* if it is *Usable and assuming Low Risk*.

Table 3
Results of applying the DMN table to a set of data records.

ID	CPU	Memory	Storage	Instance Family (DMN Output)
1	2.4	128	500 GB SSD	Memory Optimised
2	3.2	32	1000 GB	Compute Optimised
3	2.7	32	1000 GB	General Purpose
4	3.0	64	250 GB SSD	Compute Optimised

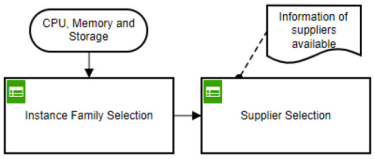


Fig. 2. Decision requirement diagram example.

3.2. Tailoring DMN elements to make the decision operative regarding data usability

Regardless of the type of business rules previously described, the formalisation of all of them is the same. The relations between them lies in the semantics derived from our conception on the hierarchy as established in the previous sections, and consequently, we will build and relate the decision table in a hierarchical structure.

3.2.1. Formalisation of data quality rules based on DMN

Based on DMN, the business rules applied in the generation of a recommendation on the potential use of a data record based on data quality concerns are formalised below. The definition of the rules include: (i) a set of input parameters; (ii) a list of if-then conditions; and (iii) the output values for each condition.

Let an instance of Business Rules for Data Decision (BR.DD) be a tuple ⟨Inputs, Rules, Outputs⟩ where:

- Inputs: This is a tuple of attributes a_i of type A_i, $\langle a_1{:}A_1, \ldots, a_n{:}A_n\rangle$, where the types permitted are String, Boolean, Integer, Real, and Date. It is represented by means of the Input data of the DMN tables.
- Output: This is a tuple of attributes b_i of type B_i, $\langle b_1{:}B_1, \ldots, b_m{:}B_m\rangle$, where the types permitted are: String, Boolean, and Integer (in a limited and finite domain). It is represented by means of the output data of the DMN tables.
- Rules: This is an ordered list of if-then rules $\langle r_1, \ldots, r_k\rangle$ where r_{i-1} has greater priority than r_i, which corresponds with the Hit Policy indicator (F) of the DMN table. Each r_i corresponds with a tuple of a DMN table, where each r_i has the form $\langle\{Q_1, \ldots, Q_n\}, \{o_1, \ldots, o_n\}\rangle$, where Q_j represents the conditions applied to the attribute a_i expressed in FEEL (if), and o_j represents the resulting *then* expression. Next, the formalisation presented in [9] is detailed:

$$Q ::= \text{“}-\text{”} \mid Term \mid \text{“}not(\text{“}\ Term\ \text{”})\text{”} \mid Comparison \mid Interval \mid Q_1,\ Q_2$$

$$Comparison ::= COpTerm$$

$$COp ::= \text{“}=\text{”} \mid \text{“}<\text{”} \mid \text{“}>\text{”} \mid \text{“}\leq\text{”} \mid \text{“}\geq\text{”} \mid \text{“}\in\text{”}$$

$$Interval ::= (\text{“(”} \mid \text{“[”})\ Term_1\ \text{“..”}\ Term_2\ (\text{“)”} \mid \text{“]”})$$

$$Term ::= v \mid f(Term_1, \ldots, Term_m)$$

For the grammar of FEEL certain remarks are needed: (i) v is a value of the domain and f is a function (e.g., +, −, round, ceiling, duration, day, etc.); (ii) “− “represents *any value*; (iii) Comparison and Interval are only applicable to numeric types; (iv) “Q_1, Q_2 “represents “$Q_1 \vee Q_2$“; and (v) if an attribute a_i fails to exist, the only condition that it could meet is “−“. For a further description of FEEL, please consult [30].

3.2.2. DMN hierarchical structure

As stated in Section 3.1, we have identified different types of BR.DD involved in the process of generating a recommendation on the usability of a data record. As stated, each BR.DD can be described by means of a decision table in DMN, and BR.DDs can be combined according to their semantics to generate a final decision regarding the data record usability. Therefore, we propose a description of all the BR.DDs as DMN tables and a combination thereof in a hierarchical way as shown in Fig. 3. The hierarchy enables: (1) BR.DV (at the top) is evaluated for every data record provided as Input. This data record can be of any of the types defined in the formalisation; (2) for each data quality dimension, a BR.DQM uses the retrieved Outputs of the required BR.DVs as Input (which can be Boolean, String or a bounded range of Integers) in a similar way; (3) a BR.DQA uses the output of different DMN tables related to the measurement of a dimension (BR.DQM) as Input; and (4) BR.DUD (at the bottom of the hierarchy) takes the Outputs of BR.DQA as Input to which it applies its if-then rules.

In addition, the output of the business rules for data decisions must return an output from an established and ordered scale, whereby the best and the worst outputs are indicated, in order to guarantee the monotonicity of the business rule [39].

4. DMN4DQ: a methodology to develop a system to generate recommendations on the usability of a data record

In order to systematically instantiate all the DMN elements identified in Section 3.1, we now introduce a methodology called DMN4DQ. DMN4DQ will guide data stewards and stakeholders towards achieving the goal of implementing a system that can be integrated along with the Information Systems supporting the business process. The methodology consists of the following phases: (i) Phase 1. Define Business Rules for Data Decisions and the underlying hierarchy (see Subsection 4.1); (ii) Phase 2. Instantiate the DMN tables of the DMN4DQ hierarchy (see Subsection 4.2); (iii) Phase 3. Deploy, test, and integrate the DMN4DQ hierarchy into the systems needing a recommendation on the potential use of a data record (see Subsection 4.3). Fig. 4 summarises these phases. We provide a detailed description of each phase in the next sections.

4.1. Phase 1. Define business rules for data decisions with the aim of generating a recommendation on data usability

The Definition of the Business Rules for Data Decisions includes the following steps aligned with the types of business rules defined in the previous section:

- Step 1.1. Define Data Context: Describe the context in which the data is used.
- Step 1.2. Describe the Dataset: Describe the dataset, its attributes, and the technological stack that supports the management and use of the data.
- Step 1.3. Define Business Rules for Data Values (BR.DV): Identify the business rules to enable the validation of the data requirements on the data to generate a value representing the extent to which the data requirement is met. All these requirements should be desirably implemented during the design of the data repository [11,28].
- Step 1.4. Select the Data Quality Dimensions that best represent the usability of the data: Identify the combination of relevant data quality dimensions that best represent business requirements for data in the specific context of the use of data, such as completeness, consistency, or any of those dimensions identified by Wang et al. [43] or ISO 25012 [21] as introduced in Section 2.1. In addition, it is necessary to identify the possible output values that can be assigned to the measurement of every data quality dimension. Although stakeholders can define any domain of values for the results of these activities, for the sake of simplicity, we propose employing Likert scales [23]. For example, the data quality dimension of consistency could admit three possible values as a result: “Sufficiently Consistent”, “Insufficiently Consistent”, and “Dramatically Non-consistent”. Additionally, in order to ensure the monotonicity of the rule [39], it is necessary to denote which value represents the highest level of quality and which represents the lowest level of quality. For example, “Sufficiently Consistent” and “Dramatically Non-consistent”, respectively.
- Step 1.5. Define Business Rules for Data Quality Measurement (BR.DQM): Identify, describe, and validate the business rules aimed to measure the chosen data quality dimensions in Step 1.4. This step needs to be broken down into two further steps: (1) to associate specific BR.DV to every data quality dimension considered; and (2) to produce the “Data Quality Measurement Business Rules” (BR.DQM) that consider the data quality requirements stated by the business data stewards for the data in a given context. Depending on the granularity, a BR.DQM can cover one or more attributes and one or more BR.DVs [34].
- Step 1.6. Define Business Rules for Data Quality Assessment (BR.DQA): Identify, describe, and validate the business rules aimed to assess the level of data quality. This step includes the following actions: (i) Identify the relative importance (i.e., weight) of the data quality dimension in the assessment of the data quality of every data record in the context of use; (ii) identify the possible states of the usability of the data (output of BR.DQA). The states that can be enumerated include: “Fully Usable”, “Usable but cleansing recommended”, “Usable with a high risk”, “Not usable”; (iii) produce the “Data Quality Assessment Business Rules” (BR.DQA) that cover the combination or aggregation of the data quality dimensions involved, and by considering their relative importance.
- Step 1.7. Define Business Rules for the generation of a recommendation on the potential usability of Data: Identify and describe the business rules aimed to generate a recommendation on the use of the data (BR.DUD) in the given context of the data. To generate a recommendation on “Using” or “Discarding” the data, it is crucial to take into account the organisational appetite-risk related to data

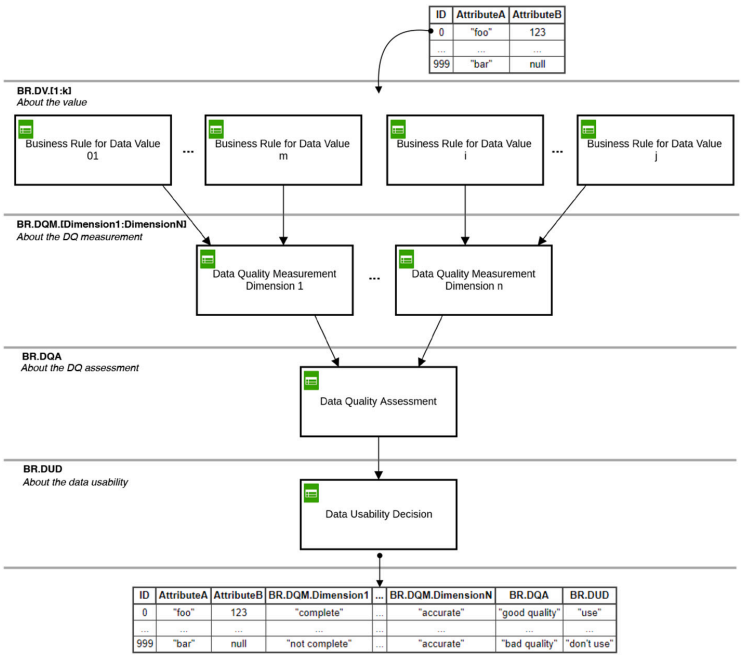


Fig. 3. Decision table diagram of DMN4DQ.

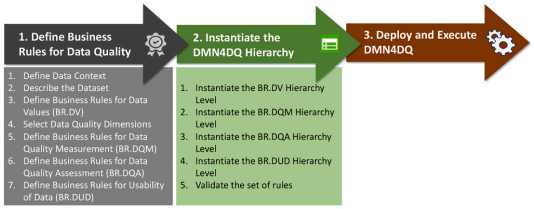


Fig. 4. The DN4DQ methodology.

quality when it comes to making a decision: in this respect, business data stewards should analyse the impact of the decision, and find a balance between discarding data or using data with a low level of quality.

4.2. Phase 2. Instantiate the DMN4DQ integration and hierarchy

As explained in Section 3.2, each set of BR.DD is represented by a DMN table that must be designed, implemented, and conveniently validated. Since there are four hierarchy levels of business rules (see Fig. 3), it is necessary to carry out the following steps: (1) Instantiate the BR.DV hierarchy level; (2) Instantiate the BR.DQM hierarchy level; (3) Instantiate the BR.DQA hierarchy level; (4) Instantiate the BR.DUD hierarchy level; and (5) Validate the set of rules as stated by D. Calvanese et al. [9].

DMN enables the decision logic to be described in a decision table. In the context of data usability recommendations, the decision table describes the data quality rules introduced by business experts that can be either correct or incorrect. A relevant previous paper [9] provides formal semantics and an algorithm for the detection of overlapping and missing rules. Other solutions can be found in the literature [41,42] but are limited to the Boolean or Enumerate domains.

4.3. Phase 3. Deploy and execute the instance of DMN4DQ decision requirement diagram

The last phase of DMN4DQ includes the development, testing, and possible deployment as an external service in a given system using software that supports an implementation of reference, as is the Camunda modeler and engine.[4]

5. Validation of DMN4DQ in a case study

The main purpose of this case study is to demonstrate that DMN4DQ can be used in a real dataset. In this case, it represents a catalogue of servers built on data provided by third parties. To ensure that the data is potentially useful in selling instances of servers in private clouds, it is necessary to analyse the data quality requirements for a decision to be made. Any lack of completeness, accuracy, and consistency of the data in this context might cause distrust among users, such as the inclusion of products in a publicity catalogue that fail to correspond to real products. In the following subsections, we show the most interesting results of this case study. The full case study is available online.[5]

5.1. Phase 1. Define business rules for recommendations on the usability of the data

Once the impact of poor-quality data in the business has been studied, the business rules for data decisions can be defined, as explained in Section 4.1.

[4] Camunda Modeler: https://camunda.com/products/modeler/
[5] DMN4Spark. Case Study: http://www.idea.us.es/dmn4dq/

5.1.1. Step 1.1. Define data context

The data, which is to be employed to build the catalogue, is extracted mainly from Amazon Web Services.[6] The data is acquired in CSV format, and each record contains information on a server instance.

5.1.2. Step 1.2. Describe the dataset

At the time of running the case study, the dataset was composed of 1,048,571 records. An extract of the data dictionary describes the dataset is: *Location* is a String attribute identifying the geographical location of the machine. It is represented by the name of the country or region where it is located; *InstanceFamily* is a String that describes the category to which the machine belongs (consistent with the features of the machine); *ClockSpeed* is a String representing the speed of the CPU (a decimal value followed by the String "GHz"); *Memory* is a String that represents the size of the RAM memory (an Integer followed by the String "GiB"); *Storage* is a String that describes the type of storage of the machine; *OperatingSystem* is a String that specifies the operating system of the machine; and *PricePerUnit* is a String representing a numeric value indicating the price of an instance of the machine.

5.1.3. Steps 1.3 and 1.4. Define business rules for data values and identify data quality dimensions

For the sake of simplicity, all the BR.DVs and the data quality dimensions to which the BR.DVs could be assimilated are presented together as follows.

Completeness. The lack of relevant data poses a potential risk in the offered service. For this specific case, we considered that the measurement of the completeness involves several BR.DVs. For the sake of simplicity, the BR.DVs are described by specifying the field to which they apply: *Location* (*BR.DV.01*), *ClockSpeed* (*BR.DV.03*), *Memory* (*BR.DV.05*), *InstanceFamily* (*BR.DV.07*), *OperatingSystem* (*BR.DV.10*), and *PricePerUnit* (*BR.DV.12*). These BR.DVs return one of the following values on a scale in the interval [0,2]: (i) 0, if the value is *null*; (ii) 1, if it is an empty String (except for BR.DV.12, for which *PricePerUnit* should be 0); and (iii) 2, otherwise. Semantic: Having a *null* value is more risky than an empty field since it might lead to misinterpretations of the idea of completeness [18]. In this case, we employ the indices of the Likert scale so that values regarding the completeness can take advantage of the operations enabled in DMN, such as the possibility of employing comparison operators in the measurement phase (see Section 3.2.1).

Accuracy. Inaccurate data might cause negative effects in terms of credibility and technical aspects. For example, if the data syntax fails to follow a specific pattern, it might not be properly processed and might cause problems when being displayed or analysed. We have considered three groups of BR.DVs involved in the measurement of the accuracy in this particular case:

1. Those which bound the value that a String can take (*BR.DV.02*, *BR.DV.08* and *BR.DV.11*, whose input fields are: *Location*, *InstanceFamily*, and *OperatingSystem*, respectively). These BR.DVs return a value on a scale composed of three elements: (i) *Appropriate* in the case where the value is in a set of very acceptable values; (ii) *Sufficiently appropriate* in the case where the value is in a set of fairly acceptable values; and (iii) *Inappropriate* if the value is not present in any set. Semantic: Unexpected values might lead to failures in data analysis processes and to misleading information. For this reason, the list of accepted values is bounded.
2. Those indicating the format which the data must take (*BR.DV.04* and *BR.DV.06*, with these inputs: *ClockSpeed* and *Memory*, respectively). They return *true* if the value matches the expected pattern. Otherwise, they return *false*. Semantic: If these fields fail to match the pattern, certain processes will fail completely.
3. Those bounding a numeric range (*BR.DV.13*, with *PricePerUnit* as input). It returns a value on a scale of three elements: (i) *Realistic* if the value is in the range (0.0, 10,000.0); (ii) *Exaggerated* if it is in the range [10,000.0, 99,999.9]; and (iii) *Unrealistic* in any other case. Semantic: Certain price values might be too high. These cases could be acceptable, but should be carefully analysed.

Consistency. Inconsistent data entails not only a potential risk from the user's point of view, but also legal issues (e.g., advertising a server instance with false characteristics). *BR.DV.09* is the only business rule we considered as necessary to be involved in the measurement of this data quality dimension, with various fields as input: *Memory*, *ClockSpeed*, *Storage*, and *IntanceFamily*. It returns a value within a range of [0,3] with the following semantic: (i) 0 if it satisfies the condition that *Memory* is less than 64 GiB and *InstanceFamily* must not be *Memory Optimised*; (ii) 1 if it satisfies the condition that *ClockSpeed* is less than 2.9 GHz and *InstanceFamily* is not *Compute Optimised*; (iii) 2 if it satisfies the condition that *Storage* does not contain the substring *SSD* and *InstanceFamily* is not *Storage Optimised*; and (iv) 3 in any other case. Semantic: Inconsistencies are more serious when found in the information regarding *Memory*, *ClockSpeed*, and *Storage*, in that order.

The next step prior to defining the BR.DQM is to design the outputs of the measurement. For the sake of simplicity, in this example we will select different Likert scales with all the possible values that might result from the measurement of each data quality dimension. We remark that it is of paramount importance to carefully study the context in which data is to be employed so that these values have a proper semantic.

Regarding the **completeness** dimension, we established the following measurement based on a Likert scale and on the risks associated to missing data: (i) *Suitably Complete* if the information about the server is complete. The record might be used in advertisement campaigns; (ii) *Sufficiently Complete* in the case where there is a minimal subset of attributes which are complete, and hence, the record can be shown in the catalogue; and (iii) *Not Complete* if the record cannot be included in the catalogue due to the lack of important attributes for sale.

Regarding the **accuracy** dimension, the management team established the following measurement levels. As in the previous case, these have been defined according to the risks associated to inaccurate data, and are based on a numerical scale: (i) 100 if the information about this record is accurate, and hence, it could be employed for advertisement campaigns; (ii) 70 in the case where there is a minimal subset of attributes which are sufficiently accurate, and hence the record could be listed in the catalogue; and (iii) 50 if values and ranges are sufficiently accurate although certain formats remain inaccurate; and (iv) 0 if there is a lack of accurate technical data, which renders this record unsuitable for listing in the catalogue.

Finally, regarding the **consistency** dimension, the following levels are defined. Again, the measurement is based on the risks associated to inconsistent data, as well as on a Likert scale: (i) *Consistent* if attributes derived from technical features are consistent between them, and hence the record could be listed in the catalogue and employed for advertisement campaigns, and (ii) *Inconsistent* if derived attributes are not consistent with technical features, the tuple must not be listed in the catalogue.

In order to simplify the proposal, other dimensions have been omitted, although their inclusion would require little effort. For example, we could have included the following BR.DV related to the timeliness dimension: the timestamp must have been generated a maximum of 15 min before the moment at which data quality measurement is performed. Its corresponding BR.DQM would set the record as *Timely* if it fulfils that BR.DV, otherwise, it would be set as *Not timely*. This BR.DV might be implemented by creating a custom function named "*current_timestamp()*", which returns the current timestamp (i.e., the timestamp at which the rule is evaluated). It would then be verified that the difference between the current timestamp and the stored timestamp is less or equal to 15 min.

[6] Dataset employed in the case study: https://www.kaggle.com/akashsarda/aws-ec2-pricing-data/version/1

5.1.4. Step 1.5. Define business rules for data quality measurement (BR.DQM)

The BR.DQM for the measurement of the **completeness** dimension (BR.DQM.Completeness) includes the following conditions: 1. A record is considered as *Suitably Complete* when the output of BR.DV.01, BR.DV.03, BR.DV.05, BR.DV.07, BR.DV.10, and BR.DV.12 are greater than or equal to 2. 2. A record is considered as *Sufficiently Complete* when BR.DV.03 and BR.DV.05 are greater than equal to 2, and BR.DV.12 is greater than or equal to 1. 3. A record is considered as *Not Complete* in any other case.

Regarding the measurement of the **accuracy** dimension (BR.DQM.Accuracy), the following conditions are defined: 1. The accuracy will have a value of 100 (*accurate*) when BR.DV.04 and BR.DV.06 are met; BR.DV.02, BR.DV.08 and BR.DV.11 are *Appropriate*, and BR.DV.13 is *Realistic*. 2. The accuracy will have a value of 70 *(sufficiently accurate)* when it meets BR.DV.04 and BR.DV.06; BR.DV.02, BR.DV.08 and BR.DV.11 are either *Appropriate* or *Sufficiently Appropriate*; and BR.DV.13 is either *Realistic* or *Exaggerated*. These records could be listed in the catalogue. 3. The accuracy will have a value of 50 when the conditions of BR.DQM.05 are met except for BR.DV.04 and BR.DV.06, which might be *false* and for BR.DV.02, which might be *Inappropriate*. 4. The accuracy will take a value of 0 otherwise.

Finally, the conditions for the business rule of the measurement of **consistency** dimension (BR.DQM.Consistency) are: 1. A record can be considered as *Consistent* when BR.DV.09 is greater than or equal to 3. 2. A record can be considered as *Inconsistent* when fails to meet BR.DV.09.

5.1.5. Step 1.6. Define business rules for data quality assessment (BR.DQA)

The output levels for the assessment have been defined as follows: (i) *Suitable or Sound Quality*. This level represents those records that are *Suitably Complete*, *Very Accurate*, and *Suitably Consistent*. The recommendations associated to these records can be to *"include them in the catalogue", "use them in advertisement campaigns"*; (ii) *Sufficient Quality*. This level represents those records that have a sufficient level of quality for them to be listed in the catalogue, although they cannot be used for advertisement campaigns to prevent risk. These records must be *Consistent* and can neither be *Not Complete* nor *Inaccurate*; and lastly, (iii) *Non-usable*. A record is *Non-usable* when it is *Not Complete*, *Inaccurate*, or *Inconsistent*. Non-usable records must not be listed in the catalogue.

The Business Rule for Data Quality Assessment is then modelled with the following conditions: 1. A record has *Suitable Quality* when its BR.DQM.Completeness is *Suitably Complete*, its BR.DQM.Consistency is *Consistent*, and its BR.DQM.Accuracy takes a value of 100. 2. A record has *Sufficient Quality* when it is *Consistent*, is not *Not Complete*, and its BR.DQM.Accuracy is greater than or equal to 70. 3. A record has *Bad Quality* when it is *Consistent*, it is not *Not Complete*, and its BR.DQM.Accuracy is greater than or equal to 50. 4. A record is *Non-usable* when it is *Not Complete*, *Inconsistent*, or its BR.DQM.Accuracy is less than 50.

5.1.6. Step 1.7. Define business rules for usability of data (BR.DUD)

This step consists of deciding the level of quality that each record from the dataset must fulfil in order to be employed in the catalogue. The decision to be made concerns whether or not to include each single record in the catalogue of server instances. According to the way in which the BR.DQA has been modelled, a record might be listed in the catalogue if its level of quality is *suitable* or *sufficient*. Thus, the conditions of the business rule for user decision-making are: 1. A record will be listed in the catalogue only when the BR.DQA is either of *suitable* or *sufficient quality*. 2. A record will not be listed in the catalogue when its BR.DQA is classified as *Non-usable*.

5.2. Phase 2. Design, implement, and validate the DMN tables

At this point, every business rule for data decisions has been modelled. Each level of the hierarchy presented in Fig. 3 must be implemented and integrated. Fig. 5 depicts the DMN hierarchy of this example. The steps followed in this example are described in the following sections.

5.2.1. Step 2.1. Instantiate the BR.DV hierarchy level

One table for each BR.DD explained in Subsection 5.1.3 must be created. Inputs are expected to be the attributes from the dataset. The output is a numeric value in the interval [0, 2] that indicates whether or not the attribute(s) fulfil(s) the conditions. The order of priority of the conditions is established when the business rule is defined, since the order of priority is in the order in which the conditions are defined. In the DMN table, each condition appears in a row in the same order in which they are defined, and the Hit Policy indicator is established as *F* (see Section 2.2). In this case study, there are 13 BR.DV, and 13 DMN tables must be created. Due to limitations on the length of the paper, only two of these BR.DVs are shown, although the reader can find the full list in the web presented by the authors.[7]

The first is BR.DV.04, depicted in Fig. 6. It is composed of three rows. The first row checks whether the input String matches the required pattern. If so, it returns the value 2. If it is an empty String, it returns the value 1, and 0 otherwise.

The second table is BR.DV.09, shown in Fig. 6. This has four inputs and four rows (if-then conditions). The three top rows are intended to verify whether the attributes *Memory*, *ClockSpeed* and *Storage* are inconsistent with the *InstanceFamily* attribute. These conditions were described in Section 5.1.3. Conditions in rows 1 and 2 verify whether the attributes *Memory* and *ClockSpeed* are less than 64 and 2.9, respectively. This is implemented by means of FEEL built-in functions. The condition is modelled by splitting the String in terms of its white spaces, then taking the expected numeric part and comparing the resulting numbers.

5.2.2. Step 2.2. Instantiate the BR.DQM hierarchy level

The DMN tables are built as described in Section 3.2. In this case, the inputs are BR.DQM are the output of the BR.DV. The measurement of each dimension is defined in a DMN table where each condition yields one value per dimension. Fig. 7 shows the DMN tables for the three defined dimensions.

5.2.3. Step 2.3. Instantiate the BR.DQA hierarchy level

Fig. 7 depicts how BR.DQA is modelled. In this case, the table inputs are the output of the business rules for data quality measurement. Each row specifies the conditions which must be accomplished for each assessment value.

5.2.4. Step 2.4. Instantiate the BR.DUD hierarchy level

Fig. 7 depicts the modelling of the BR.DUD. The input is the result of the BR.DQA.

5.2.5. Step 2.5. Validate the set of rules

DMN tables may be validated [9]. We propose the use of two tools to validate the DMN tables: dmn-js,[8] which verifies a table by checking possible missing and overlapping rules; and dmn-check,[9] which checks duplicate rules, conflicting rules, shadowed rules, types of expressions, correct use of enumerations, and correctly connected requirement graphs.

5.3. Phase 3. Deploy, test, integrate, and execute the tables obtained by applying DMN4DQ

We developed a tool, called *dmn4spark*,[10] which takes a DMN file and

[7] DMN4DQ: http://www.idea.us.es/dmn4dq/
[8] dmn-js: http://dmn.cs.ut.ee.
[9] dmn-check: https://github.com/red6/dmn-check#validations.
[10] dmn4spark: https://github.com/IDEA-Research-Group/dmn4spark

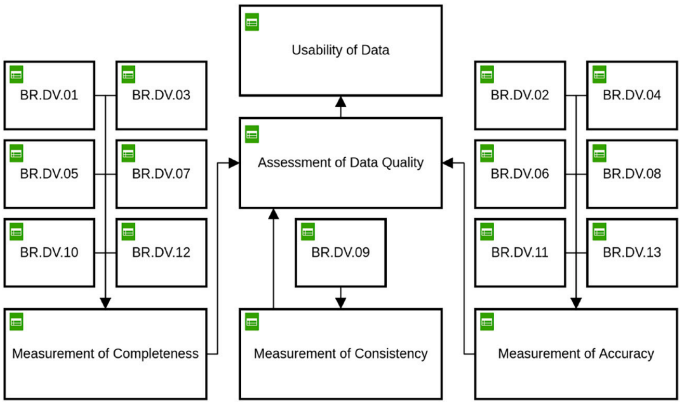


Fig. 5. DMN4DQ - Decision table diagram of the case study.

Validation of BR.DV.04

Validation_BR.DV.04

F	Input	Output
	ClockSpeed	BR.DV.04
	String	Number
1	matches(ClockSpeed, "^(\d+(?:\.?)\d* GHz)$")	2
2	ClockSpeed == ""	1
3	-	0

Validation of BR.DV.09

Validation_BR.DV.09

F	Input				Output
	Memory	ClockSpeed	Storage	InstanceFamily	BR.DV.09
	String	String	String	String	Number
1	number(if(Memory != null) then split(Memory, " +")[1] else null) < 64	-	-	"Memory optimised"	0
2	-	number(if(ClockSpeed != null) then split(ClockSpeed, " +")[1] else null) < 2.9	-	"Compute optimised"	1
3	-	-	not(contains(Storage, "SSD"))	"Storage optimised"	2
4	-	-	-	-	3

Fig. 6. DMN tables for BR.DV.04 and BR.DV.09.

a dataset as inputs, and evaluates all the DMN tables for each record of the whole dataset. This tool is based on Apache Spark,[11] a distributed computing framework. In this way, users can obtain a recommendation for the usability of each data record of the dataset in a given context in Big Data scenarios. One of the main advantages of Apache Spark is the fact that it abstracts users from defining data models, since it is able to infer the schema of semi-structured datasets. In addition, this tool offers the possibility of using external plugins for the structuring of datasets by means of data transformation techniques [40]. Once the corresponding DMN file is defined, it must be uploaded to HDFS or a web server reachable by the cluster on which the application will be run. The steps to follow to use this tool are summarised in Fig. 8.

We employed this implementation to compute the results for the dataset of the case study in order to generate a recommendation on the potential usability of each of the 1,048,571 records. The results are depicted in Tables 4, 5, 6, 7, and 8. These show, for each DMN table, the number of tuples and the percentage thereof which fulfil each possible result of the business rules.

5.4. Conclusion about the execution of the case study

Regarding the results obtained, several conclusions can be drawn: (i) For almost half of the records the recommendation to discard them has been generated. This means that the potential risk of not having filtered out the data which fails to meet minimum standards of quality could have been much higher, since the existence of defects in the definition of half of the server instances would have strongly deteriorated the quality of the services offered, and consequently the reputation of the Company. If the organisation wants to increase the number of usable records, then the quality of the data must be improved; (ii) The main root cause of low-quality data is the lack of accuracy, given that around 37% of the records have an accuracy in the range of [0, 50]. These records might should therefore be analysed in order to find the root cause of the inaccuracy; and (iii) intermediate cases such as *sufficiently complete*, *accurate*, and *sufficient quality* are not very common.

[11] Apache Spark: http://spark.apache.org/

Measurement of Completeness — Decision_Completeness

F	Input: BR.DV.01	BR.DV.03	BR.DV.05	BR.DV.07	BR.DV.10	BR.DV.12	Output: BR.DQM.Completeness
	Number	Number	Number	Number	Number	Number	{Suitably Complete, Sufficiently Complete, Not Complete}
1	>=2	>=2	>=2	>=2	>=2	>=2	Adequately Complete
2	-	>=2	>=2	-	-	>=1	Complete Enough
3	-	-	-	-	-	-	Not Complete

Measurement of Consistency — Decision_Consistency

F	Input: BR.DV.09	Output: BR.DQM.Consistency
	Number	{Consistent, Inconsistent}
1	>=3	Consistent
2	-	Inconsistent

Measurement of Accuracy — Decision_Accuracy

F	Input: BR.DV.02	BR.DV.04	BR.DV.06	BR.DV.08	BR.DV.11	BR.DV.13	Output: BR.DQM.Accuracy
	{Appropriate, Sufficiently Appropriate, Inappropriate}	Boolean	Boolean	{Appropriate, Sufficiently Appropriate, Inappropriate}	{Appropriate, Sufficiently Appropriate, Inappropriate}	{Realistic, Exaggerated, Unrealistic}	Number
1	Appropriate	true	true	Appropriate	Appropriate	Realistic	100
2	not(Inappropriate)	true	true	not(Inappropriate)	not(Inappropriate)	not(Unrealistic)	70
3	-	-	-	not(Inappropriate)	not(Inappropriate)	not(Unrealistic)	50
4	-	-	-	-	-	-	0

Assessment of Data Quality — Decision_Assessment

F	Input: BR.DQM.Completeness	BR.DQM.Accuracy	BR.DQM.Consistency	Output: BR.DQA
	{Suitably Complete, Sufficiently Complete, Not Complete}	Number	{Consistent, Inconsistent}	{Suitable Quality, Sufficient Quality, Bad Quality, Non-usable}
1	Adequately Complete	100	Consistent	Suitable Quality
2	not(Not Complete)	>=70	Consistent	Sufficient Quality
3	not(Not Complete)	>=50	Consistent	Bad Quality
4	-	-	-	Non-usable

User Decision Making — Decision_Making

F	Input: BR.DQA	Output: BR.DUD
	{Suitable Quality, Sufficient Quality, Bad Quality, Non-usable}	{Use, Do not use}
1	Suitable, Sufficient Quality	Use
2	-	Do not use

Fig. 7. DMN tables for the Completeness, Accuracy, and Consistency dimensions; the Assessment and the Data Usability Decision.

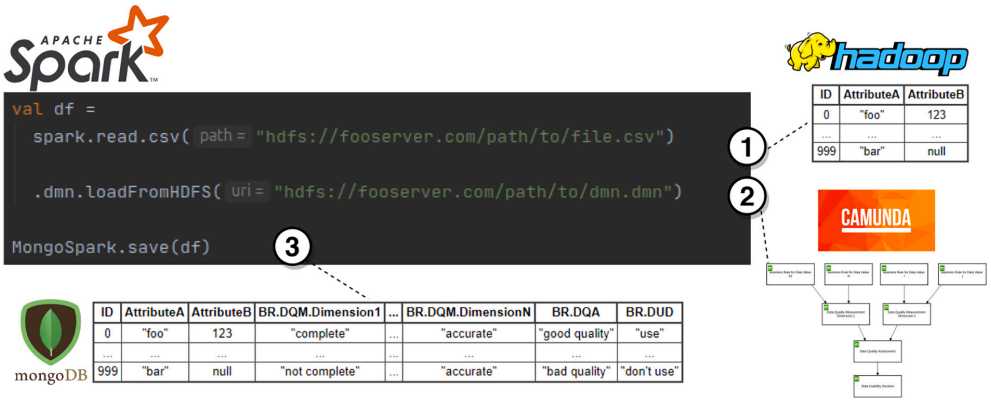


Fig. 8. Steps to follow for using our tool *dmn4spark*.. In this example, the dataset and the DMN file are stored in HDFS, and the results are dumped in MongoDB.

Table 4
Ratio of results for the measurement of completeness dimension.

BR.DQM.Completeness	#	%
Suitably Complete	839,990	80.11
Sufficiently Complete	888	0.08
Not Complete	207,692	19.81

Table 5
Ratio of results for the measurement of accuracy dimension.

BR.DQM.Accuracy	#	%
100	629,106	60.00
70	25,022	2.39
50	392,714	37.45
0	1728	0.16

Summarising, the generation of a recommendation on the usability of the data helps both to automate the data quality assessment and the detection of the reason why the data fails to satisfy the business rules defined. Therefore, an in-depth study into the quality of those records which have been considered non-usable should be carried out.

6. Related work

Organizations today are aware of the importance of ascertaining the levels of the quality of data. The necessity to generate recommendations on the use of the data records based on some business restrictions with

Table 6
Ratio of results for the measurement of the consistency dimension.

BR.DQM.Consistency	#	%
Consistent	909,565	86.74
Inconsistent	139,005	13.26

Table 7
Ratio of the results for data quality assessment.

BR.DQA	#	%
Suitable	520,271	49.62
Sufficient Quality	25,022	2,39
Bad Quality	185,862	17.73
Non-usable	317,415	30.27

Table 8
Ratio of generated recommendations.

BR.DUD	#	%
Use	545,296	52.00
Do not use	503,277	48.00

regard to the measured or assessed level of data quality (i.e., the appetite for risk involved in using data with inadequate levels of quality) has been studied previously [14,29,36,45]. However, DMN4DQ goes one step further in that it is a holistic solution where the processes of measurement, assessment (these two typically considered as synonyms), and generation of a recommendation of the use of data are integrated and adequately related by incorporating the business needs. Furthermore, we have tailored the OMG's international standard DMN to support the automation of the required actions to generate the recommendation on the potential use of data grounding our proposal on the concept of decision rules. To model the decision rules about data can be described, we ground our proposals on previous works aimed to formalise the data quality rules [27,34], expressed through some business rules [2] that data should meet. Other proposals, as [39], reflect that the discovering and definition of business rules - expressed by regular expressions, representing functional dependencies, by using control digits or employing association analysis - constitute the cornerstone of any data quality management initiative [1,13,17,34]. However, it is important to highlight that our work is not about discovering and defining business rules, but to combine them to generate automated business-based recommendation. In this sense, we encourage to read and to use the works describing traditional types of integrity constraints for data quality management, such as functional dependencies (FDs), and their extension conditional functional dependencies (CFDs) [5,16] or even the Fellegi-Holt method that automatically "corrects" data that fail some predefined requirements [6]. On the other hand, there exist generic approaches to define business rules but not used in the context of data quality as used in the paper. For example, SBVR [31] facilitates the definition of vocabularies and rules, but it is not decision-oriented. As said, to make operational the integration of the different parts, and to facilitate the modelling of the decision rules, instead of proposing a new one language, we propose the application of DMN, the OMG's standard, which includes the FEELs. FEELs increments the easiness and feasibility of the writing of the rules - and the agnostic-technological implementation. Some other authors have proposed their own frameworks to automatically measure the levels of data quality, just to name a few, let us bring the works done by Liu et al. who introduces in [26] a semantic-aware data quality assessment for image big data; or the work by [3] who propose a methodology to build a data quality adapter module selecting the best configuration for the data quality assessment in big data. However, to the best of our knowledge, DMN4DQ is the first solution that integrates every type of decisions needed to judge about the usability of a data record in the same framework, being our contribution the tailoring of such mechanisms to support a holistic solution.

7. Limitations of the proposal

In this section, we analyse the potential limitations of the proposals. Firstly, our approach is thought to be applied record by record (e.g. acting on a given tuple). Consequently, the definition of the rules is thought to describe business restrictions applying to every record, not to the whole dataset. However, the generalisation would not be difficult by including some logic aimed at computing global measurement on the whole datasets, which was initially out of the scope of our investigation. And secondly, the main issues of validation according to [46] are of internal, external and conclusion validity. 1. *Internal validity* refers to the trustworthiness of the result. In this respect, our work can be limited to three lines: (a) the assessment and measurement processes are database and data type agnostic, but are carried out over each independent tuple; (b) the type of business rules is limited to the support currently provided by the DMN specification and the FEEL language; and (c) the assessment and measurement of complex dimensions could require additional effort to construct auxiliary and extra functions in order to obviate complex attributes and rules. 2. *External validity* refers to the generalisation and the potential interest in the approach. To encourage the validation, usability, and generalisation of our approach: (a) we have provided a methodology; (b) we have provided a tool; and (c) a step-by-step case study is given and results of the tool are analysed. Therefore, researchers or practitioners who wish to use, replicate, or extend our approach are welcome to do so. 3. *Conclusion validity* refers to the rigorousness in the relationship established between the research questions raised and the findings obtained. We have striven to overcome this limitation by providing all the resources employed in the paper, namely, the tool and the data used in pursuit of repeatability and replicability of the findings established.

8. Conclusions

The usability of the data largely depends on the data quality, and on the context where the data is used. In this paper, we have presented a methodology that integrates different types of business rules for data decisions, holistically tackling the data-usability recommendation. DMN4DQ provides a hierarchy to integrate decision rules about data values, measurements of various dimensions, assessment through the aggregation of dimensions, and the data usability. Derived from the necessity to make decisions regarding the data usability, we rely on the OMG standard for decisions, DMN, as a suitable mechanism to model and automate the generation of the recommendations on the usability of the data in a specific context, since it coherently and comprehensively enables the description and evaluation of the business rules regarding data decisions. Moreover, DMN facilitates the transformation of the knowledge held by business experts into a formal model. Thanks to the use of DMN, the automation of the evaluation of the level of data quality ceases to be a solely theoretical contribution and becomes real technology that is applicable to real scenarios. Furthermore, we have developed a tool that supports the methodology validated with a real dataset.

Author statement

All the authors are responsible for the concept of the paper, the results presented and the writing. All the authors have approved the final content of the manuscript. No potential conflict of interest was reported by the authors.

Acknowledgements

This work has been partially funded by the Ministry of Science and

Technology of Spain via ECLIPSE (RTI2018-094283-B-C33 and RTI2018-094283-B-C31) projects; the Junta de Andalucíavia the COPERNICA and METAMORFOSIS projects; the European Fund (ERDF/FEDER); the Junta de Comunidades de Castilla-La Mancha via GEMA: Generation and Evaluation of Models for dAta quality (Ref.: SBPLY/17/180501/000293), and by the Universidad de Sevilla with *VI Plan Propio de Investigación y Transferencia* (VI PPIT-US).

References

[1] Z. Abedjan, L. Golab, F. Naumann, Profiling relational data: a survey, VLDB J. 24 (2015) 557–581.
[2] P. Alpar, S. Winkelsträter, Assessment of data quality in accounting data with association rules, Expert Syst. Appl. 41 (2014) 2259–2268.
[3] D. Ardagna, C. Cappiello, W. Samá, M. Vitali, Context-aware data quality assessment for big data, Futur. Gener. Comput. Syst. 89 (2018) 548–562, https://doi.org/10.1016/j.future.2018.07.014.
[4] C. Batini, C. Cappiello, C. Francalanci, A. Maurino, Methodologies for data quality assessment and improvement, ACM Comput. Surv. (CSUR) 41 (2009) 1–52.
[5] P. Bohannon, W. Fan, F. Geerts, X. Jia, A. Kementsietsidis, Conditional functional dependencies for data cleaning, in: 2007 IEEE 23rd International Conference on Data Engineering, 2007, pp. 746–755.
[6] A. Boskovitz, R. Goré, M. Hegland, A logical formalisation of the fellegi-holt method of data cleaning, in: M. Berthold, H.J. Lenz, E. Bradley, R. Kruse, C. Borgelt (Eds.), Advances in Intelligent Data Analysis V, Springer Berlin Heidelberg, Berlin, Heidelberg, 2003, pp. 554–565.
[7] A. Bronselaer, R. De Mol, G. De Tré, A measure-theoretic foundation for data quality, IEEE Trans. Fuzzy Syst. 26 (2017) 627–639.
[8] I. Caballero, E. Verbo, C. Calero, M. Piattini, A Data Quality Measurement Information Model Based on iso/iec 15939, ICIQ, Cambridge, MA, 2007, pp. 393–408.
[9] D. Calvanese, M. Dumas, Ü. Laurson, F.M. Maggi, M. Montali, I. Teinemaa, Semantics and analysis of dmn decision tables, in: M. La Rosa, P. Loos, O. Pastor (Eds.), Business Process Management, Springer International Publishing, 2016, pp. 217–233.
[10] F. Chiang, R.J. Miller, Discovering data quality rules, in: Proceedings of the VLDB Endowment 1, 2008, pp. 1166–1177.
[11] E.F. Codd, Extending the database relational model to capture more meaning, ACM Trans. Database Syst. 4 (1979) 397–434, https://doi.org/10.1145/320107.320109.
[12] T. Davenport, J. Harris, Competing on Analytics: Updated, with a New Introduction: The New Science of Winning, Harvard Business Press, 2017.
[13] L. Ehrlinger, E. Rusz, W. Wöß, A Survey of Data Quality Measurement and Monitoring Tools, 2019 arXiv preprint arXiv:1907.08138.
[14] A. Even, G. Shankaranarayanan, Utility-driven assessment of data quality, in: ACM SIGMIS Database: The DATABASE for Advances in Information Systems 38, 2007, pp. 75–93.
[15] A. Even, G. Shankaranarayanan, P.D. Berger, Evaluating a model for cost-effective data quality management in a real-world crm setting, Decis. Support. Syst. 50 (2010) 152–163.
[16] W. Fan, Data quality: Theory and practice, in: H. Gao, L. Lim, W. Wang, C. Li, L. Chen (Eds.), Web-Age Information Management. WAIM 2012. Lecture Notes in Computer Science vol. 7418, Springer, Berlin, Heidelberg, 2012, pp. 548–553.
[17] W. Fan, Data quality: from theory to practice, ACM SIGMOD Rec. 44 (2015) 7–18.
[18] W. Fan, J. Li, S. Ma, N. Tang, W. Yu, Towards certain fixes with editing rules and master data, VLDB J. 21 (2012) 213–238.
[19] K. Figl, J. Mendling, G. Tokdemir, J. Vanthienen, What we know and what we do not know about DMN, Enterpr. Model. Inform. Syst. Architect. 13 (2) (2018) 1–16.
[20] B. Glavic, Big data provenance: Challenges and implications for benchmarking, in: Specifying Big Data Benchmarks - First Workshop, WBDB 2012, San Jose, CA, USA, May 8-9, 2012, and Second Workshop, WBDB 2012, Pune, India, December 17-18, 2012, Revised Selected Papers, 2012, pp. 72–80, https://doi.org/10.1007/978-3-642-53974-9_7.
[21] ISO-25012, Iso/iec 25012: Software Engineering-Software Product Quality Requirements and Evaluation (square)-Data Quality Model, 2008.
[22] ISO-25024, Iso/iec 25024:2015 Systems and Software Engineering — Systems and Software Quality Requirements and Evaluation (square) — Measurement of Data Quality, 2015.
[23] A. Joshi, S. Kale, S. Chandel, D.K. Pal, Likert scale: explored and explained, Br. J. Appl. Sci. Technol. 7 (2015) 396.
[24] J. Ladley, Data Governance: How to Design, Deploy, and Sustain an Effective Data Governance Program, Academic Press, 2019.
[25] S. Lee, B. Ludäscher, B. Glavic, PUG: a framework and practical implementation for why and why-not provenance, VLDB J. 28 (2019) 47–71.
[26] Y. Liu, Y. Wang, K. Zhou, Y. Yang, Y. Liu, Semantic-aware data quality assessment for image big data, Futur. Gener. Comput. Syst. 102 (2020) 53–65.
[27] D. Loshin, 17 - data quality and business rules in practice, in: D. Loshin (Ed.), Enterprise Knowledge Management, Academic Press, San Diego. The Morgan Kaufmann Series in Data Management Systems, 2001, pp. 425–461, https://doi.org/10.1016/B978-012455840-3.50017-0.
[28] D. Loshin, The practitioner's Guide to Data Quality Improvement, Elsevier, 2010.
[29] J. Merino, I. Caballero, B. Rivas, M.A. Serrano, M. Piattini, A data quality in use model for big data, Future Generation Comp. Syst. 63 (2016) 123–130, https://doi.org/10.1016/j.future.2015.11.024.
[30] OMG, Decision Model and Notation (DMN), Version 1.2, URL, https://www.omg.org/spec/DMN, 2019.
[31] OMG, Semantics Of Business Vocabulary And Rules, Version 1.5, URL, https://www.omg.org/spec/SBVR/About-SBVR/, 2019.
[32] B. Otto, Y.W. Lee, I. Caballero, Information and data quality in business networking: a key concept for enterprises in its early stages of development, Electron. Mark. 21 (2011) 83.
[33] J.M. Pérez-Álvarez, A. Maté, M.T. Gómez-López, J. Trujillo, Tactical business-process-decision support based on kpis monitoring and validation, Comput. Ind. 102 (2018) 23–39, https://doi.org/10.1016/j.compind.2018.08.001.
[34] L. Sebastian-Coleman, Measuring Data Quality for Ongoing Improvement: A Data Quality Assessment Framework, Newnes, 2012.
[35] K. Sha, S. Zeadally, Data quality challenges in cyber-physical systems, J. Data Inform. Qual. (JDIQ) 6 (2015) 1–4.
[36] G. Shankaranarayanan, Y. Cai, Supporting data quality management in decision-making, Decis. Support. Syst. 42 (2006) 302–317, https://doi.org/10.1016/j.dss.2004.12.006.
[37] V.C. Storey, R.M. Dewan, M. Freimer, Data quality: setting organizational policies, Decis. Support. Syst. 54 (2012) 434–442, https://doi.org/10.1016/j.dss.2012.06.004.
[38] J.G. Suarez, in: Philip B. Crosby, W. Edwards Deming, Joseph M. Juran (Eds.), Three Experts on Quality Management, Total Quality Leadership Office, Arlington VA, 1992. Technical Report.
[39] Y. Timmerman, A. Bronselaer, Measuring data quality in information systems research, Decis. Support. Syst. 126 (2019) 113138, https://doi.org/10.1016/j.dss.2019.113138.
[40] Á. Valencia-Parra, Á.J. Varela-Vaca, M.T. Gómez-López, P. Ceravolo, Chamaleon: Framework to Improve Data Wrangling with Complex Data, in: 40th International Conference on Information Systems, ICIS 2019, Association for Information Systems, 2019. URL, https://aisel.aisnet.org/icis2019/data_science/data_science/16.
[41] J. Vanthienen, E. Dries, Illustration of a decision table tool for specifying and implementing knowledge based systems, Int. J. Artif. Intell.Tools 3 (1994) 267–288.
[42] J. Vanthienen, C. Mues, A. Aerts, An illustration of verification and validation in the modelling phase of KBS development, Data Knowl. Eng. 27 (1998) 337–352, https://doi.org/10.1016/S0169-023X(98)80003-7.
[43] R.Y. Wang, A product perspective on total data quality management, Commun. ACM 41 (1998) 58–65.
[44] R.Y. Wang, M.P. Reddy, H.B. Kon, Toward quality data: an attribute-based approach, Decis. Support. Syst. 13 (1995) 349–372, https://doi.org/10.1016/0167-9236(93)E0050-N (information technologies and systems).
[45] S. Watts, G. Shankaranarayanan, A. Even, Data quality assessment in context: a cognitive perspective, Decis. Support. Syst. 48 (2009) 202–211, https://doi.org/10.1016/j.dss.2009.07.012.
[46] C. Wohlin, P. Runeson, M. Höst, M.C. Ohlsson, B. Regnell, Experimentation in software engineering, Springer. (2012), https://doi.org/10.1007/978-3-642-29044-2.

Álvaro Valencia-Parra (PhD. Student) Universidad de Sevilla, Dpto. Lenguajes y sistemas informáticos – Spain. Álvaro Valencia-Parra obtained his B.S degree in Software Engineering at the University of Seville in 2017. In 2019, he graduated with honors from the University of Seville with a M.Sc. degree in Computer Engineering. Currently, he is a PhD student. His research areas include the improvement of different activities in the Big Data Pipeline, such as data transformation, data quality, and data analysis. The scenarios he is facing up are mainly focused on the process mining paradigm. Hence, his goal is to improve the way in which final users deal with data preparation and specific scenarios in which configuring a Big Data Pipeline might be tricky. For this purpose, he is working in the improvement of these processes by designing Domain-Specific Languages, user interfaces, and semi-automatic approaches in order to assist users in these tasks. He has participated in prestigious congresses such as the BPM Industry Forum or the International Conference on Information Systems (ICIS).

Luisa Parody (Associate Professor), Universidad Loyola Andalucía, Sevilla, Spain. Luisa Parody studied computer engineering (including a minor in systems engineering) at the Universidad de Sevilla (Spain) and graduated with honors in July 2009. She then earned an M.Sc.degree in software engineering and technology(2010) and obtained her international PhD with honors at the Universidad Sevilla (2014). Since 2018, she has been working as an associate professor in Dto. Método Cuantitativos at the Universidad Loyola. She belongs to the IDEA Research Group and has participated in several private and public research projects and has published several high-impact papers.

Ángel Jesús Varela-Vaca, (Assistant Professor) Universidad de Sevilla, Dpto. Lenguajes y sistemas informáticos – Spain. Angel J. Varela-Vaca received the B.S. degree in Computer Engineering at the University of Seville (Spain) and graduated in July 2008. M.Sc. on Software Engineering and Technology (2009) and obtained his PhD with honors at the University of Seville (2013). Angel is currently working as Assistant Professor at Languages and System Informatics Department at the Universidad Sevilla and belongs to the Idea Research Group. Angel has and leaded various private projects and participated in several public research projects and he has published several impact papers. He was nominated as a member of Program Committees such as ISD 2016, BPM Workshops 2017, SIMPDA 2018. He has been reviewer for international journals such as Journal of Supercomputing, International Journal of Management Science and Engineering Management Multimedia Tools and Applications, Human-Centric Computational and Information Sciences, Mathematical Methods in Applied Sciences among others.

Ismael Caballero, (Associate Professor) Universidad de Castilla-La Mancha, Dpto. Tecnologías y Sistemas de Información. ISMAEL CABALLERO received the M.Sc. and Ph.D. degrees in computer science from the University of Castilla-La Mancha, Spain, in 2004, where he works as Associate Professor with the Information Systems and Technologies Department. In 2017, he cofounded the spinoff DQTeam where he serves as Training Head. He has been researching on data quality management and data governance, since 1998, coauthoring several books, conference and journal articles. He teaches data quality management and data governance foundations in many universities and companies. He holds several professional certifications: CISA certification by ISACA, since 2016, CDO-1 certification by UALR-MIT, since 2017. He is currently a member of ISO TC184/SC4 working as Project Editor for several parts of ISO 8000-60 series development project. He led the project of ISO 8000-62SMAEL CABALLERO received the M.Sc. and Ph.D. degrees in computer science from the University of Castilla-La Mancha, Spain, in 2004, where he works as Associate Professor with the Information Systems and Technologies Department. In 2017, he cofounded the spinoff DQTeam where he serves as Training Head. He has been researching on data quality management and data governance, since 1998, coauthoring several books, conference and journal articles. He teaches data quality management and data governance foundations in many universities and companies. He holds several professional certifications: CISA certification by ISACA, since 2016, CDO-1 certification by UALR-MIT, since 2017. He is currently a member of ISO TC184/SC4 working as Project Editor for several parts of ISO 8000-60 series development project. He led the project of ISO 8000-62.

María Teresa Gómez-López, (Associate Professor) Universidad de Sevilla, Dpto. Lenguajes y sistemas informáticos – Spain. María Teresa Gómez-López is a Lecturer at the University of Seville and the head of the IDEA Research Group. Her research areas include Business Processes and Data management, and how to improve the business process models including better decisions and enriching the model with Data Perspectives. She has led several private and public research projects and has published several impact papers, among others in Information and Software Technology, Information Systems, Information & Software Technology, or Data & Knowledge Engineering. She was nominated as a member of Program Committees, such as ER, BPM, EDOC, ISD or CAISE Doctoral Consortium. She has been reviewing for international journals, such as International Journal of Data and Information Quality, Journal of Systems and Software, Artificial Intelligence in Medicine; Business & Information Systems Engineering Journal or Information Science. She has been invited as a keynote speaker in various occasions, such as at the Workshop on Data & Artifact Centric BPM, International Workshop on Decision Mining & Modelling for Business Processes BPM, the X National Conference of BPM, the 28th IBIMA Conference.

3.2.3 Unleashing Constraint Optimisation Problem solving in Big Data environments

Published in the Journal of Computational Science (Vol. 45, p.101180). Elsevier BV.

- ***Authors**: Álvaro Valencia-Parra, Ángel Jesús Verala-Vaca, Luisa Parody, María Teresa Gómez-López.*
- ***DOI**: 10.1016/j.jocs.2020.101180.*
- ***Rating**: Q2 (JCR'20 3.976).*

Journal of Computational Science 45 (2020) 101180

Contents lists available at ScienceDirect

Journal of Computational Science

journal homepage: www.elsevier.com/locate/jocs

Unleashing Constraint Optimisation Problem solving in Big Data environments

Álvaro Valencia-Parra[a], Ángel Jesús Varela-Vaca[a,*], Luisa Parody[b], María Teresa Gómez-López[a,1]

[a] Dpto. Lenguajes y Sistemas Informáticos, Universidad de Sevilla, IDEA Research group, Spain
[b] Dpto. Métodos Cuantitativos, Universidad Loyola Andalucía, IDEA Research group, Spain

ARTICLE INFO

Article history:
Received 25 November 2019
Received in revised form 2 June 2020
Accepted 18 June 2020
Available online 24 June 2020

Keywords:
Big Data
Optimisation problem
Constraint programming
Distributed data
Heterogeneous data format

ABSTRACT

The application of the optimisation problems in the daily decisions of companies is able to be used for finding the best management according to the necessities of the organisations. However, optimisation problems imply a high computational complexity, increased by the current necessity to include a massive quantity of data (Big Data), for the creation of optimisation problems to customise products and services for their clients. The irruption of Big Data technologies can be a challenge but also an important mechanism to tackle the computational difficulties of optimisation problems, and the possibility to distribute the problem performance. In this paper, we propose a solution that lets the query of a data set supported by Big Data technologies that imply the resolution of Constraint Optimisation Problem (COP). This proposal enables to: (1) model COPs whose input data are obtained from distributed and heterogeneous data; (2) facilitate the integration of different data sources to create the COPs; and, (3) solve the optimisation problems in a distributed way, to improve the performance. It is done by means of a framework and supported by a tool capable of modelling, solving and querying the results of optimisation problems. The tool integrates the Big Data technologies and commercial solvers of constraint programming. The suitability of the proposal and the development have been evaluated with real data sets whose computational study and results are included and discussed.

1. Introduction

The use of optimisation problems let organisations manage the resources, time, and cost of their processes. However, the necessity to create customised products according to the profiles of the clients implies the creation of thousands of optimisation problems. Moreover, the incorporation of more interesting data, frequently provided by multiple systems and services [33], will make companies more competitive. Moreover, the integration of more complex data implies the resolution of optimisation problems that could suppose a computationally complex task, further an extra effort to integrate heterogeneous data format provided by different sources. COPs are frequently used to model and solve optimisation problems since they include among their advantages: the ability to model problems declaratively, regardless of how it will be solved; their applicability to many real-life examples of industry, so its versatility and power are empirically demonstrated; and the existence of a significant amount of commercial tools that solve the constraint satisfaction problems locally. The resolution time of the problem can be compromised for both the complexity of the constraints and the size of the data set. Thereby, in the current scenarios, the information involved in a COP is not always centralised. The exponential growth of the data produced and stored and the distribution of the information have promoted the use of Big Data paradigm [25,31], producing difficulties that have not been adapted to COPs [41]. This new situation can produce that, the data and constraints that model their relations are distributed among different subsystems (also known as nodes) that constitute the global model. These models can be computationally highly complex since they can have distributed data or constraints. This interruption of Big Data into COPs can be seen as a new challenge, but also as a new opportunity to solve distributed problems and data.

Before the Big Data burst into the scene, Distributed Constraint Optimisation Problems (DCOPs) [21,11] were defined to solve COPs where neither their constraints nor data were in a single system.

* Corresponding author.
E-mail addresses: avalencia@us.es (Á. Valencia-Parra), ajvarela@us.es (Á.J. Varela-Vaca), mlparody@uloyola.es (L. Parody), maytegomez@us.es (M.T. Gómez-López).
[1] http://www.idea.us.es/

https://doi.org/10.1016/j.jocs.2020.101180
1877-7503/

DCOP paradigm proposes a set of algorithms to solve a set of COPs that share variables. The resolution of each of them must be aware of the rest of the constraints that restrict the values of the variables. For this reason, DCOPs are focused on the communication between the distributed nodes that must synchronise the share variables to obtain a correct result of every constraint, variable and in each node. However, DCOP proposals do not provide a mechanism for a better distribution of the DCOPs for the optimal resolution of them. In this context, our proposal is based on the creation and distribution of isolated COPs in a Big Data infrastructure, whose partial resolution provides information to solve a greater problem utilising the MapReduce paradigm [10]. Furthermore, none of the algorithms proposed to solve DCOP have been presented as a practical and appropriate solution to actual environments, even less when the data used is characterised by its high variability, velocity, and volume (i.e., Big Data). Although the existence of the problem was formulated in our previous works [35,36], to the best of our knowledge, there are no solutions that adequate the COPs to data supported by Big Data infrastructure, dealing with large amounts of data, using the distributed node to parallelise the computation, getting results quickly, and managing a variety of data efficiently.

As aforementioned, we identified the necessity of the COP resolution within different Big Data scenarios in [35,36]. In those previous work, we focused on describing the scenarios and their characteristics, then a solution was prototyped. However, the previous work lacks formalisation, implementation and evaluation with real data was done.

In order to fulfil those gaps, we propose a solution which enables to define, execute and solve Constraint Optimisation Problems with Distributed Data in Big Data environment. The main contributions are four main:

- **Formalisation of Constraint Optimisation Problems with Distributed (Big) Data**: An optimisation problem must be modelled describing the constraints that relate the data and the objective function to be optimised. When the data is distributed and the resolution of the problems depends on this data, the description of the model must include the distributed data than can have heterogeneous formats and come from Big Data environments. We formalise COP models that include these aspects.
- **Integration and transformation of heterogeneous data formats**: The heterogeneity of the data formats makes necessary data preparation by means of a transformation into a unified format to apply the latter analysis. This task is especially relevant in our proposal since the mapping between input data to the optimisation problem must be defined, at the same time that the output obtained must be recovered.
- **Enable the optimal evaluation of queries over data provided by the resolution of Optimisation Problems**: The resolution of various COPs can provide useful information, for example, to ascertain the most suitable products for each provider. However, to select the most appropriate for a specific organisation, these partial solutions obtained from the evaluation of the COPs must be queried. Our proposal is focused on the analysis of the queries over the COP data output to avoid, when it is possible, the resolution of the COPs, minimising the evaluation time.
- **Provide an integral framework to support the process**: To cover the previous proposals, it is necessary a leap of advances in technologies that support the integration of the Big Data infrastructures with constraint optimisation solvers. This new technology is proposed as a new component in the Big Data ecosystem to enable the management of a great amount of data and the creation and distribution of several COPs, whose output data can be later queried.

Unfortunately, there are no technological solutions that enable the application of optimisation problems with Big Data characteristics in real scenarios. This lack of tools means that these challenges need an extra effort to be solved.

In order to tackle our proposal, FAst BIg cOstraint LAboratory (FABIOLA) framework is proposed. FABIOLA is a new Hadoop-based component [3] in the Big Data ecosystem. FABIOLA enables the modelling of optimisation problems whose heterogeneous format of the data, the amount of data to deal with, and the required velocity in the resolution of the problems forces a Big Data solution. The FABIOLA framework also provides the necessary techniques to solve COPs in a distributed way, either to obtain a result by optimising time and resources or because the nature of the problem is distributed.

FABIOLA prototype was presented in previous works [35,36], thereby, this paper can be seen as an extension of the previous ones. However, certain challenges tackled here were not tackled in that previous works: how to transform the data for its integration with other distributed data; how to optimise the data obtained from the optimisation by means of a query language that reduces the COPs evaluation, and; how an integral solution could support the whole process providing an implemented tool used for the community as a new component of the Big Data ecosystem, and; the empirical evaluation of the approach with real data.

The remainder of the paper is organised as follows: Section 2 formalises the modelling of the optimisation problems in Big Data environments. Section 3 details the different phases of the proposal that are illustrated with a running example to make the problem understandable. Section 4 analyses various operators for processing the results obtained from the resolution of the optimisation problems. Section 5 details the proposed architecture and the methodology necessary to support FABIOLA framework and the developed infrastructure. Section 6 presents the formalisation applied to a real case with a high volume of data. This section also shows a computational, statistical, and analytical study of our proposal. Section 7 includes the main related work. Finally, conclusions are drawn and future work is proposed in Section 8.

2. Overview of the proposal: Optimisation Problems within Big Data scenario

The creation of a *COPDD* is a complex task, especially when the quantity of data involved, and the number of COPs is very high. This is why this creation must be recommendable automated. For this reason, our proposal integrates a set of components that facilitate both the creation and the distribution of the COPs for their resolution.

Definition 1. A COPDD is defined by the tuple $\langle$DS, COP, DM, $\text{DMap}_{\text{DS}\rightarrow\text{DM}}$, $\text{DMap}_{\text{DM}\rightarrow\text{COP}}\rangle \rightarrow \text{DS}^{output}$, where:

- DS is the Data Set used as input data to find out the optimal solution of the problem.
- COP is the Constraint Optimisation Problem.
- DM is the Data Model that describes the relationship between the attributes of the DS and the COP.
- $\text{DMap}_{\text{DS}\rightarrow\text{DM}}$ and $\text{DMap}_{\text{DM}\rightarrow\text{COP}}$ are the Data Mapping (DMap) that represents the relation between the attributes of the DS and the attributes of the DM, and the attributes of the DM and the variables used in the COP, respectively.
- DS^{output} is a data set obtained from the resolution of an optimisation problem according to the attributes defined in the DM and their corresponding values.

Each element of the tuple of the COPDD plays a role to create and solve the COPs in an efficient way, as shown in Fig. 1. They are combined to create a framework that integrates the four phases of the

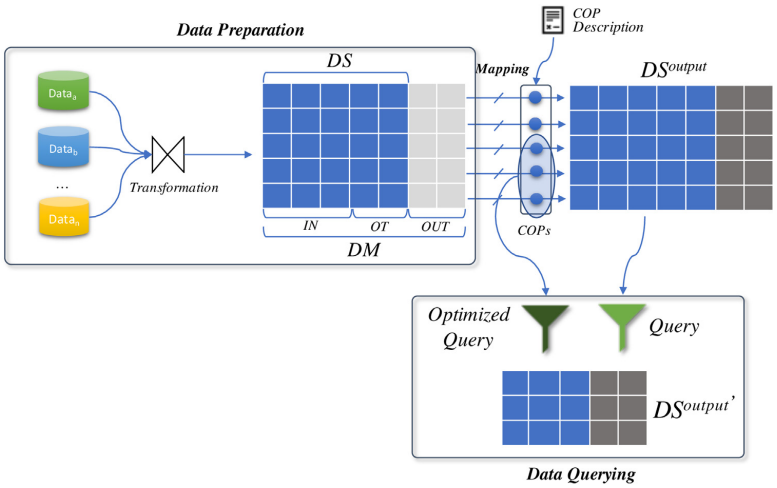


Fig. 1. Overview of the approach.

COPDD creation: (1) data set preparation; (2) COP description; (3) mapping among the data sets and the COP, and; (4) data querying optimising the performance. Some definitions have been included to clarify the phases of the proposal, each of them is detailed in the following sections.

2.1. Running example

To understand the proposal, a simple running example is used, although a real and more complex case study to compute the evaluation is given in Section 6. The running example is about a planning problem [9], that describes a company that manufactures two types of products: a standard product A, and a more sophisticated product B. Management charges a certain price for each unit of product A and a price for the unit of product B, whose profits are p_A and p_B respectively. Therefore, the *profit* of the company comes from the multiplication of the price by the number of units (i.e., q_A and q_B for products A and B) of each product. Manufacturing one unit of product A requires h_A hours of labour and rm_A units of raw material. Similarly, for one unit of B, h_B hours of labour and rm_B units of raw material are needed. Besides, the company's policy indicates that the number of units manufactured of product A has to be, at least, the double of units than the produced for product B. At present, a nH of hours of labour and a nRM of units of raw material are available. The problem is focused on to *maximise* the company's total revenue. Although every element of the COPDD will be described in the following sections, a brief approximation to the formalisation with the running example can be done at this point by identifying the Data Set (DS) and data to optimise:

- DS: $price_A$, $price_B$, $numUnits_A$, $numUnits_B$, $hours_A$, $hours_B$, $hoursLabour$, raw_A, raw_B, $unitsRawMaterial$.
- COP is defined to maximise the *profit*.

3. Phases of the proposal

As presented in Fig. 1, four are the phases that our proposal supports to create and solve the COPDDs. Following sections describe each of them.

3.1. Data set preparation

Data preparation is one of the most consuming processes in the Big Data pipeline [22,44]. One of the steps is the data transformation to fit the heterogeneous data formats into a single one for a later application of algorithms over all tuples. This type of problem is also tackled for the COPDD, that needs a DS with homogeneous tuples used as inputs. It implies that each input data ($Data_a$, $Data_b$, ..., $Data_n$) must be transformed into a single one structure (DS), following the data transformation processes as detailed in [46]. Therefore, a heterogeneous data format is a set of data ($Data_a, Data_b, \ldots, Data_n$) formed of a different set of attributes among them.

Definition 2. A DS is a set of tuples formed of a set of attributes, $\{a_1, a_2, \ldots, a_n\}$, where each tuple presents an assignment of values $\{val_1, val_2, \ldots, val_n\}$ to each attribute.

Some of the elements of the second row of Table 1 (cf., $\#ID$, nH, nRM, h_A, rm_A, p_A, h_B, rm_B, p_B) presents the elements of the DS for the running example.

3.2. Constraint Optimisation Problem description

A COP is a Constraint Satisfaction Problem (CSP) where an optimisation function determines the 'best solution' from the set of possibles. They come from Constraint Programming (CP) [39], an Artificial Intelligence (AI) discipline where a large number of problems and other areas of Computer Science can be seen as individual cases of CSP. Examples include scheduling, temporal reasoning, graph problems, and configuration problems. The basis of CP stands on the resolution of a CSP. Thereby, to understand correctly what is a COP, the CSP definition must be introduced before.

Definition 3. A CSP represents a reasoning framework consisting of variables, domains and constraints $\langle V, D, C \rangle$, where V is a set of n variables $\{v_1, v_2, \ldots, v_n\}$ whose values are taken from finite domains $\{D_{v_1}, D_{v_2}, \ldots, D_{v_n}\}$ respectively, and C is a set of constraints on their values. The constraint $c_k(x_{k_1}, \ldots, x_{k_m})$ is a predicate that is defined on the Cartesian product $D_{k_1} \times \ldots \times D_{k_j}$. This predicate is true iff the value assignment of these variables satisfies the constraint c_k.

Table 1
Example of tuples for the running example.

	IN								OUT			OT
#ID	*nH*	*nRM*	h_A	rm_A	p_A	h_B	rm_B	p_B	***profit***	q_A	q_B	Comp.
#1	1500	400	3	4	5	4	6	10	570	58	28	Comp1
#2	900	200	1	2	3	2	3	4	300	100	0	Comp2
#3	1200	–	7	4	14	–	7	30	160	80	4640	Comp2
#4	–	300	2	3	9	4	6	11	900	100	0	Comp3
#5	1500	–	–	1	8	–	–	15	12000	1500	0	Comp4

Table 2
COP for the running example.

Variables &domains:	$price_A$, $price_B$, *profit*: Float; $numUnits_A$, $numUnits_B$, $hours_A$, $hours_B$: Integer; raw_A, raw_B, *hoursLabour*, *unitsRawMaterial*: Integer;
Constraints:	$\sum_{i=A}^{B}(hours_i * numUnits_i) \leq hoursLabour$ $\sum_{i=A}^{B}(raw_i * numUnits_i) \leq unitsRawMaterial$ $numUnits_A - 2 * numUnits_B \geq 0$ $profit = \sum_{i=A}^{B}(price_i * numUnits_i)$
Optimisation function:	*Maximise*(*profit*)

The search for solutions for a CSP is based on the instantiation concept. An assignment of a variable, or instantiation, is a pair variable-value (x, a) which represents the assignment of the value a to the variable x. An instantiation of a set of variables is a tuple of ordered pairs, where each sorted pair (x, a) assigns the value a to the variable x. A tuple $((x_1, a_1), \ldots, (x_i, a_i))$ is consistent if it satisfies all the constraints formed by variables of the tuple.

Several possible solutions can be found to satisfy a CSP, but sometimes only the more suitable wants to be obtained, one that optimises the solution. In those cases, a COP must be modelled.

Definition 4. A COP is a CSP in which the solutions optimise (minimise or maximise) an objective function, *f*.

For the example, for maximising the company's total revenue can be formulated as shown in Table 2.

The values of *nH*, *nRM*, h_A, rm_A, p_A, h_B, rm_B, p_B depend on the company where the products are built. Table 1 shows some possible scenarios, where the output data (i.e., *profit*, q_A, q_B) is obtained according to each input data (per tuple). Every input data can be instantiated (e.g., tuples # 1 and # 2), and the output indicates the resulting *profit* and how many units of each product produces (cf., q_A and q_B). It is also possible that some input values were unknown (cf., the value '–' in the table). In that case, the COP tries to find values for these variables to optimise the variable *profit* (cf., tuples # 3, # 4 and # 5). In this example, we can observe that there is another attribute, such as *Company* (cf., *Comp.*), that is not part of and does not influence the resolution of the optimisation problem. However, this kind of variables can be helpful to answer later queries, such as, *Which is the manufacturer with the highest profit? Which is the manufacturer that produces more products B?*

If there are several tuples of input data, several are the possible optimal solutions that can be found. Thereby, the created COP can be seen as a meta-COP which is partially instantiated for each tuple of the DS. How to map the attributes of DS with the inputs of the COP is described is explained in the next subsections.

3.3. The mapping between the DS and COP

In order to link the DS with the COP, it is necessary to define an independent DM which allow users to define how the data from the DS is used as input of the COPs, and the resultant data obtained by solving the COPs.

Definition 5. A DM is a set of attributes, $\{a_1, a_2, \ldots, a_n\}$, which is divided into three disjointed groups: Input (IN), Output (OUT), and Others (OT) attributes.

$$IN \cap OUT = \emptyset \wedge IN \cap OT = \emptyset \wedge OT \cap OUT = \emptyset \quad (1)$$

These sets of attributes are defined as:

- *IN:* specific attributes from the DS that will be linked to the input variables of the COP.
- *OUT:* attributes that describe the output data of the COP. The values of these attributes are obtained from the COP resolution.
- *OT:* specific attributes form the DS that describe additional information and that can be used to make further queries combining them with output attributes. These attributes are unrelated to the COP since they lack influence in its resolution.

The DM enables the separation of the DS and the COP model in such a way that the COP model can be applied to many data sets. However, to relate the DM and attributes of the DS and the variables of the COP, we need to introduce another concept, the DMap.

Following definitions determine how the DS attributes and the COPs inputs are mapped through the DM.

Definition 6. $\text{DMap}_{\text{DS}\to\text{DM}}$ is the relation between the attributes of DS and the IN and OT attributes of the DM, formed of a list of attributes $\{a'_i, \ldots, a'_m\} \subseteq$ DS where a subset of attributes are related to IN, $\{a'_i, \ldots, a'_m\} \mapsto \{IN\}$, and another subset of attributes are related to OT, $\{a'_h, \ldots, a'_g\} \mapsto \{OT\}$.

$$\text{DMap}_{\text{DS}\to\text{DM}} : \text{DS} \mapsto \{IN, OT\}, \quad (2)$$

$$\forall a'_i \in \text{DS} : \exists i_x \in IN | a'_i = i_x \quad \wedge \quad \nexists o_h \in OT | a'_i = o_h, \quad |IN| \geq 1 \quad (3)$$

Definition 7. $\text{DMap}_{\text{DM}\to\text{COP}}$ is the relation between the attributes of the data model DM and the variables of the COP, *V*. It is described by a list of attributes $\{i_k, \ldots, i_n, out_g, \ldots, out_l\}$ where a subset of attributes from IN are related to the input variables of the COP, $\{i_k, \ldots, i_n\} \mapsto \{V\}$, and another subset of attributes from OUT are related to the output variables of the COP, $\{out_g, \ldots, out_l\} \mapsto \{V\}$.

$$\text{DMap}_{\text{DM}\to V} : \{IN, OUT\} \mapsto V, \quad |V| \geq 2 \quad (4)$$

$$\forall i_k \in IN : \exists v_j \in V | i_k = v_j \quad \wedge \quad \nexists v_h \in V | i_k = v_h \quad (5)$$

$$\forall out_g \in OUT : \exists v_i \in V | out_g = v_i \quad \wedge \quad \nexists v_j \in V | out_g = v_j, \quad (6)$$

$$\forall (i_k, out_g) | i_k \in IN, out_g \in OUT : \nexists v_t \in V | i_k = v_t \quad \wedge \quad out_g = v_t \quad (7)$$

Following the running example, the DS in Table 1 is remarked as IN, OT, and OUT according to the DM definition and to represent the DMap. To illustrate how the Data Mapping is carried out, once the transformation has been performed. Fig. 2 represents the mapping between the DS and the COP through the DM. A first DMap determines the relation between the attributes of the DS and the DM. The second DMap relates the DM and the variables of the COPs [46]. Applied to the running example, the attributes of DS (*nH*, *nRM*, h_A, rm_A, p_A, h_B, rm_B, p_B, *Comp.*) are mapped into attributes of the DM as labelled as IN and OT in Table 2. For instance, *nH* from the DS is mapped as *nH* IN's attribute. Similarly, *nH* IN's attribute is mapped as *hoursLabour* variable of the COP.

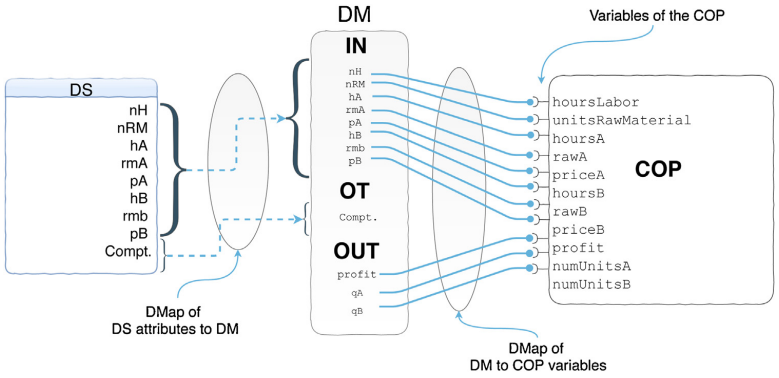


Fig. 2. Data mapping applied to the example.

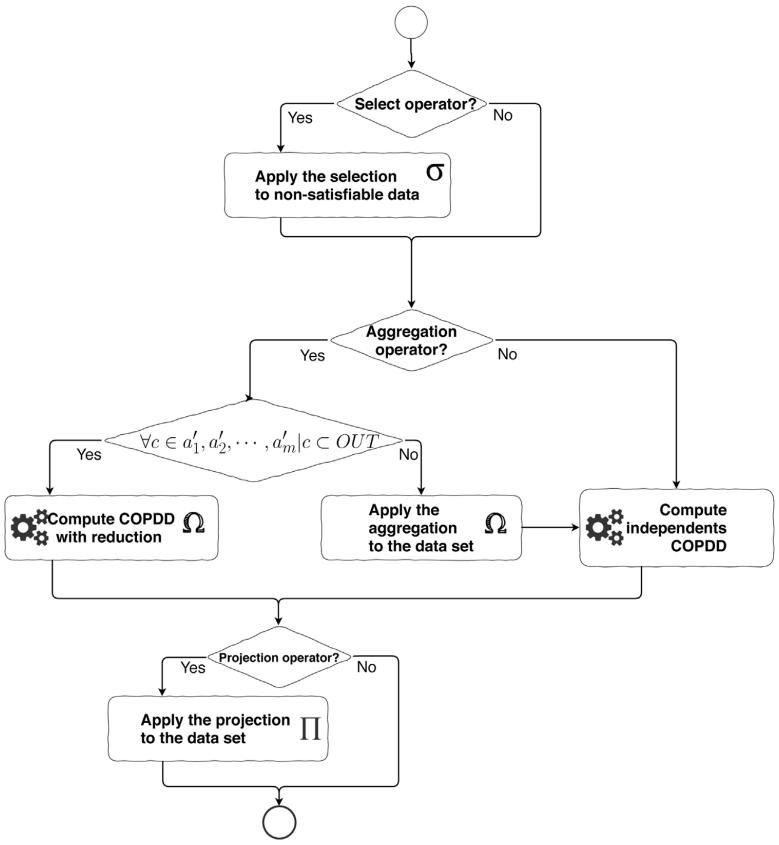


Fig. 3. Process for computing COPDD for distributed data.

Regarding the constraints, the COP can be modelled employing a set of fixed constraints such as seen in the example of Table 2. Therefore, each tuple represents a possible assignment of data to the variables of the COP, which can be solved (i.e., OUT) independently. However, in other types of problems, the constraints are built dynamically due to the constrained-relation existing between the values of the attributes of the DM. Thus, each tuple produces a customised COP. The solution of these customised COPs could be influenced by a criterion regarding the IN attributes, for instance, to order the COPs based on descending criteria of p_A attribute. This

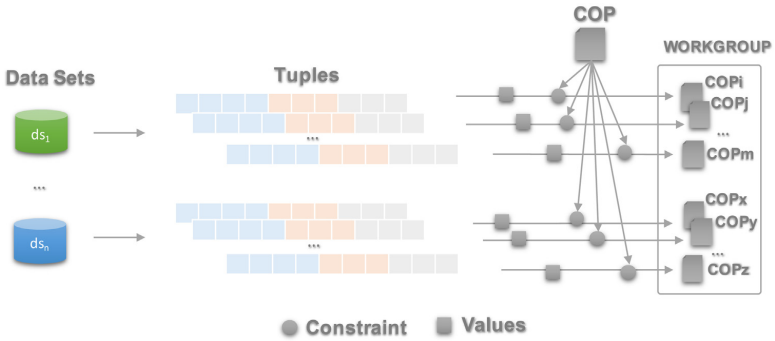


Fig. 4. Illustration for independent resolution of COPDDs.

criterion establishes the solution of *ID* tuples: # 3, # 4, # 5, # 1, and # 2. Thus, COP for the tuple # 3 will be solved in the first place, the COP for the tuple # 4 in the second place, and so on.

3.4. Data querying

The DS^{output} contains every DS tuple combined with the outputs (OUT) obtained after the distributed COPs evaluations. Later analysis of this data should be relevant. Following the running example, to know the maximum *profit*, the minimum or the average. Classical query operators can be applied to DS^{output}, but it is necessary to analyse which types of attributes (IN, OUT and OT) can be involved in each operator. For this reason, we revisit the classical relational algebra [42] employing its unary operators (i.e., selection, projection and aggregation) that can be applied to DS^{output}.

- Selection (σ). When the selection operator is applied to a DS^{output}, depending on the types of attributes involved in the φ, the selection can be applied before and/or after the resolution of the COPDD. In our proposal, only IN and OT attributes can be involved.
- Projection ($\prod$). When the projection operator is applied to a DS^{output}, the attributes involved $\{a_1, a_2, \ldots, a_n\} \subset \{\text{IN, OUT, OT}\}$. For this reason, the projection is applied after the optimisation problem resolutions, since OUT attributes can be included in the projection.
- Aggregation (Ω). There are five aggregation functions (f), i.e., *SUM*, *COUNT*, *AVG*, *MAX*, and *MIN*. The application of each function returns a number (Integer or Float) and can be applied in IN, OUT and OT attributes.

These three operators can be combined and nested (thanks to the closure property of the relational algebra) to query a DS^{output}. In following section, we propose to overtake part of the query evaluation to the COPs resolution to reduce the number of optimisation problems to solve, and take the advantages of the distributed and parallel evaluation of the COPs, as it is detailed in the following subsections.

4. Optimising data set query for solving COPs

The presented operators can involve every tuple, however, some data analysis done by querying could not entail every tuple, such as 'to obtain the maximum *profit* when *nH* is greater than 1, 200'. In this section, we analyse how they can be solved to reduce the computation complexity of obtaining and querying DS^{output}, proposing the systematic process shown in Fig. 3 applied before the COPDDs evaluation, since the order in which the operators are applied affects the time consuming of the distributed resolution of the optimisation problems drastically, being crucial how the tasks to solve a query are scheduled [43] in Big Data ecosystems. The application of these steps avoids the resolution of COPs that do not fit the selection operator or those that do not correspond with the attribute to aggregate.

The **first step** is to analyse whether a **selection operator** is included. It might reduce the input data involved in the COPDDs. For instance, in the running example of the previous section, let be only optimised the products whose rm_A have a value greater than 2, the tuples # 2 and # 5 might not be included in the computation of the COPDDs. If there is a selection operator, the application of the filtering must be developed, as it is carried out in classical data. In the current proposal, only selections over input and other types of attributes can be applied.

The **second step** is related to the **aggregation operator**. If an aggregation operator is included, how the information is managed depends on the types of attributes and the aggregation functions (i.e., *SUM*, *COUNT*, *AVG*, *MAX*, and *MIN*). IN and OT are managed as usual data applying the aggregation operator or relational algebra, meanwhile OUT for *MIN* or *MAX* function will be used to improve the optimisation of the COPs to reduce the search space of the variables. Therefore, the **Computation of independent COPDD** or **Computation COPDD with reduction** will be carried out depending on whether an aggregation operator over OUT attributes have been used or not. Both types of COPDDs are explained in the following subsections. The **third step** is related to the **projection operator**, where a new DS is created as a subset of attributes of the DS^{output}. This new DS is the result of the projection operator and the solution to the problem. An example is to obtain the *profit* but not being necessary to return q_A and q_B.

4.1. Computation of independent COPDD

When operators are applied to OUT attributes, every COPs can be solved individually, being or not in distributed nodes. Thereby, the DS described in the formalisation can be divided into different and distributed nodes. DS can be represented as a set of data sets, $ds_1, ds_2, \ldots, ds_n$, each of them located in an independent node. Each tuple p within a $ds_i \in \{ds_1, ds_2, \ldots, ds_n\}$ represents a problem to be optimised. Each COP depends on the values of the DM for each tuple. It is similar to use each tuple p to instantiate the COP, creating a COP_p in which the values are taken from p. The group of COPs generated for each tuple of all DS conforms a *workgroup*, as we defined in Fig. 4. A particularity of this scenario is that the COPs within the *workgroup* can be computed independently without the application of any operator. Thus, the *workgroup* can be seen as a unique task

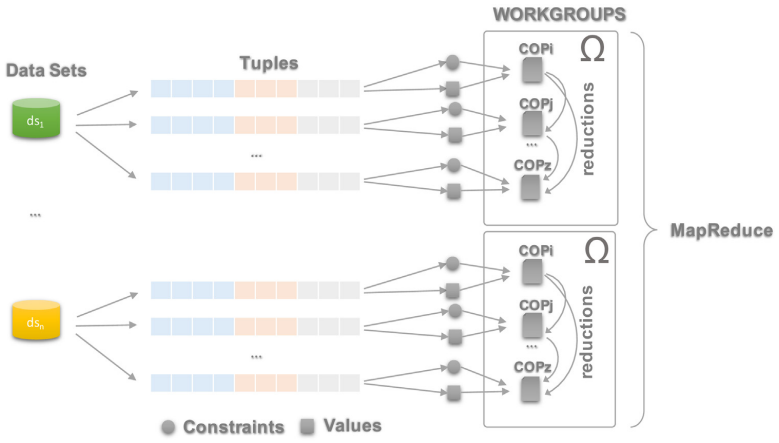


Fig. 5. Illustration for reduction based on aggregation.

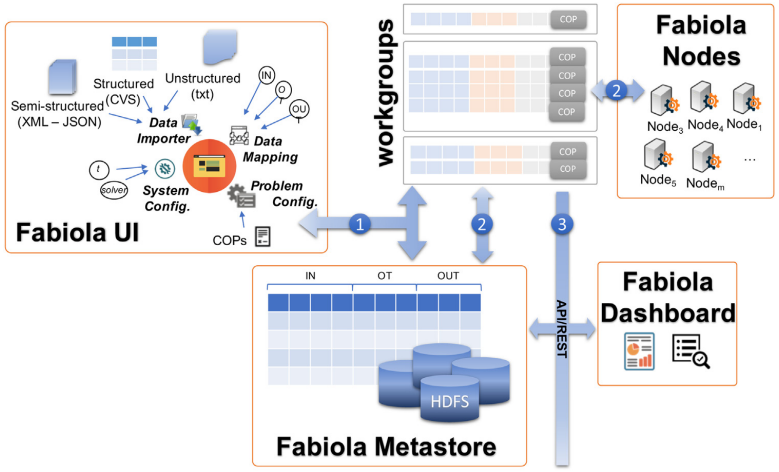


Fig. 6. Architecture and methodology of FABIOLA.

that needs to be computed in a distributed way. As commented, in the running example can be to obtain the average of the *profit* for all the products, being necessary to compute all the COPs to obtain the *profit* values before obtaining the average. The calculation of the *profit* for each COP is unrelated to any other COP. Therefore, the computation of the COPs can be performed independently at any order, because all the COPs must be solved.

4.2. Computation of COPDD with reduction

When an aggregation operator is applied to obtain the *MAX* or *MIN* value of an OUT attribute, the evaluation of a COP_p can use solutions obtained from the resolution of previous COPs of the same DS (ds_i). Thus, each tuple conforms a COP_p in which constraints and values are taken directly from p. There are possible values from the same DS that might be shared from one solved COP_i to another unsolved COP_j to improve the search for solutions reducing the search space. Thus, the domain of variables of the (unsolved) COP_j might be reduced to enhance the search space. Following the running example, the objective function could be to determine the maximum *profit* with the minimum value of units of A (cf., q_A). In this case, assuming the solution of the COP, for instance, for the tuple #1 the values of *profit* and q_A (cf., Table 1) could be used to reduce the domains of the variables *profit* and $numUnits_A$ in the unsolved COPs since a greater value of $numUnits_A$ and a less value of *profit* than previously calculated are unacceptable. Despite reduction operations, the aggregation operator (i.e., Ω) is necessary since the results of all COPs must be combined to determine the maximum *profit* and the minimum q_A. As defined in Fig. 3, the *aggregation operation* has to be applied in the output variables *profit* and $numUnits_A$.

In this scenario, problems from the same DS might be related internally to each other by *reduction operations* but they are unrelated to problems from other DSs. Therefore, the COPs of each DS, ds_i, conforms independent *workgroups* that can be solved without regarding other *workgroups*. Thus, each *workgroup* can be seen as a unique task that needs to be computed in a distributed way. In Fig. 5, two *workgroups* are depicted related to the DS, ds_1 and ds_n,

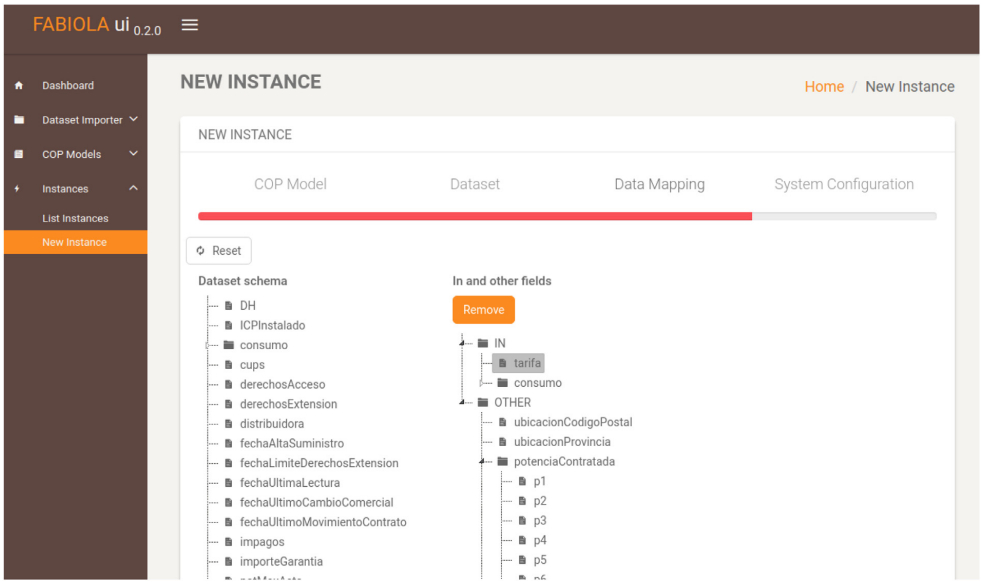


Fig. 7. Data mapping in FABIOLA-UI.

wherein each *workgroup* reduction are illustrated as arrow transitions between the COPs and Ω represents the aggregation operator which is applied in this context.

In Big Data environments, the resolution of a problem can be based on partial problem resolutions that are combined, which is known as MapReduce technique [10]. It can also be applied to the resolution of optimisation problems. For instance, the maximum value of an attribute, where the value of this is distributed in different *workgroups*, is the maximum value of the all maximum obtained for each *workgroup*. This implies to optimise a problem, with the optimisation of each *workgroup*, that has been solved in proposals such as Hive [45]. The question is, how this optimisation process can be affected when the partial resolution implies also the COP resolutions. The solution of all *workgroups* need to be combined (cf. Fig. 5) to generate a global solution as a new OUT. A way of combining needs to be defined over the OUT attributes of COPs. In order to carry out this, MapReduce for optimisation is defined as follows.

Definition 8. A MapReduce for Optimisation Problems, *MR*, is defined as a combination function which establishes the maximum or minimum value of an OUT_i variable, being $OUT_i \in \{out_1, out_2, \ldots, out_n\}$ and establishes a way of the combination by means of the operator, OP, as minimise (*MIN*) or maximise (*MAX*).

Each variable $v \in \{out_1, out_2, \ldots, out_n\}$ involved in the aggregation operator for a COP_i is bounded with a specific value obtained in previous solved COPs of the same DS, (ds_j). This specific value is a maximum whether the *MIN* value wants to be obtained, or the minimum whether the *MAX* function is used in the aggregation.

5. FABIOLA: the technological approach

FABIOLA is presented as a technological solution for the automatic creation of COPs with distributed Data. In this section, the needed infrastructure, including the architecture, and the implemented tool are detailed.

FABIOLA as a technological solution is composed of two differentiated parts: (1) the infrastructure (i.e., back-end) for the computation, and; (2) a web interface (i.e., front-end) to enable the access to the components described in previous sections. The FABIOLA back-end and front-end components are depicted in Fig. 6.

The FABIOLA front-end is divided into two components:

- **FABIOLA-UI** component is a web client-side application which enables to setup the environment to compute COPDDs. It provides sections for the *Data importer* to load external data; the COP description; Data Transformation; the *Problem configuration* (i.e., *Data Mapping*), and the *System Configuration* in terms of solver, memory, timeout of execution, etc.
- **FABIOLA Dashboard** is a control panel from where the user can use various reporting and querying components. The dashboard enables users an easy-querying and visualisation of the data and results in FABIOLA.

FABIOLA-UI, on the left, provides a fixed menu which gives direct access to different parts of FABIOLA components such as the dashboard, DS importers, DS creation, COP models, and instances as shown in Fig. 7. The same figure presents the main view once entering FABIOLA-UI in the section of creating a new instance. The creation of a new instance in FABIOLA means the creation step-by-step of a new COPDD. The figure shows an example of how to carry out the data mapping between the DS (cf., DS schema) and the attributes of the DM. The user can choose by drag-and-drop which attributes from imported DS are IN and OT.

FABIOLA-UI provides an editor for the description of the COP model in Constraint Optimisation Problem. Currently, FABIOLA-UI supports any ChocoSolver-based optimisation syntax. Although the description of the COP can be a difficult task, it is done once and can be applied to every DS. In a similar way than in data mapping between DS and DM, the DMap between DM and the variables of the COP is described.

FABIOLA-UI provides customised interfaces where operators can be applied over the data and COP results. For instance, *projection operations* can be applied for the results such as the forms provided. In this case, it is described the attributes to projection, selection and

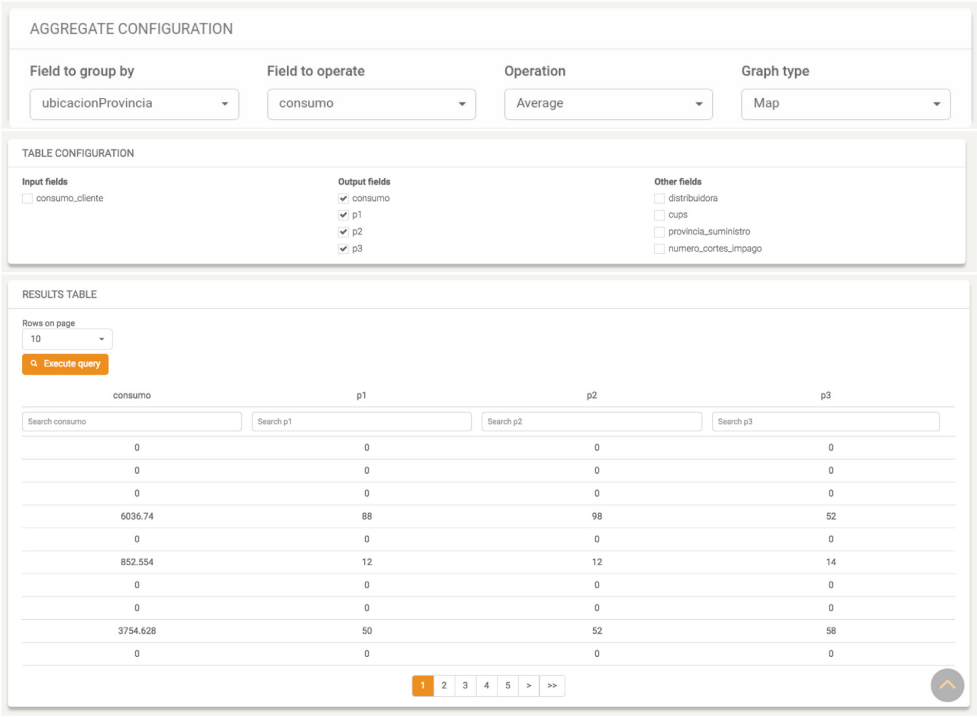


Fig. 8. Data visualisation of results.

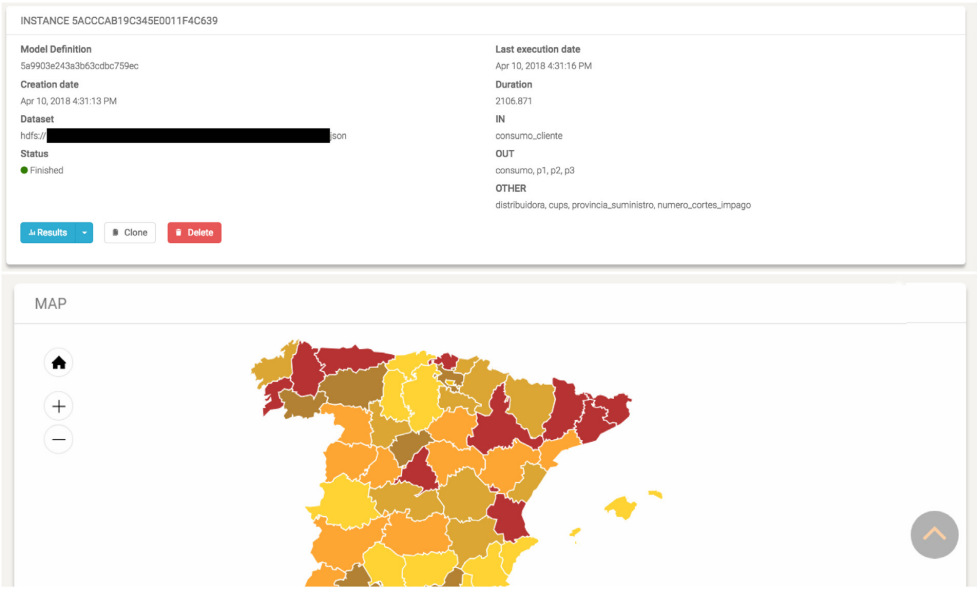


Fig. 9. Data visualisation aggregated by regions.

aggregation, as shown in Fig. 8 respectively. This use of operators will permit the view of the data, enabling an easy data visualisation and analytics where grouping and aggregated operators can be applied for filtering the results.

Other ways of presenting, grouping and aggregating of results can be defined. Fig. 9 presents an example of a dashboard for showing the results and the information about the computation of the COPDD. In this case, the results have been ranged in regions and located in a geographical map since FABIOLA Dashboard lets the extension by using modules to present data results.

The FABIOLA back-end is composed of the following components:

- **FABIOLA Metastore** represents the internal data storage mechanism of FABIOLA. It presents the problems in a structured way, where data is organised in the form of tables and schemes, such as in Hive [45]. These tables and schemes are only a virtual representation (i.e., view) of the data since it is internally organised in the original format of a distributed file system, for instance, HDFS or NFS.
- **FABIOLA Nodes** are responsible to compute the COPs. Thus, each row (tuple) instantiates a COP, and the generated COPs are grouped into *workgroups*. These *workgroups* are uniformly allotted to be solved among the available nodes and according to the workload of the nodes.

The FABIOLA back-end has been implemented by an Apache Spark [49] cluster managed by DC/OS.[2] The cluster consists of (cf., Fig. 10) a master node, responsible for planning the execution and managing the cluster resources, and N agent nodes, which are responsible for managing the applications deployed on the cluster. In our case, the cluster is composed of five agent nodes for running Apache Spark. One of these nodes is the driver, responsible for creating and distributing the tasks. These are executed by the four Spark workers. Additionally, the architecture includes a server with HDFS (cf., HDFS [3]) to store the DSs, and a database (cf., MongoDB [30]) for storing execution and partial results. Regarding computational characteristics, each node from the cluster reaches four cores and 16 gigabytes of main memory.

6. Computational experiments in a real scenario

In this section, the application of FABIOLA on a real case with and without the data set query optimisation is applied (cf., Section 4) is illustrated in depth.

6.1. Case of study: electric energy consumption scenario

In Spain, seven wholesale electricity companies are responsible for supplying power to all users. However, more than 250 distributors are responsible for selling power directly and invoice for it. Customer acquisition is, undoubtedly, the most important target that the distributors have. This acquisition determines the success or failure of the company. If an electricity retailer optimises its customers' invoice, its client portfolio will increase. The optimisation of the invoice includes the customisation and improvement according to the specific client consumption. In this case, this scenario is focused on the optimisation and customisation of the consumption of each point of supply but using the rates and prices established by the Ministry of Industry. In this scenario, it is necessary to build a COP, which is going to be applied to each of the specific values of consumption of each of the existing supply points (more than 20,000,000 points in Spain). The computer processing is even more difficult since the information provided by the seven wholesalers varies in format and is distributed in various servers.

Although the format of the data sets provided is heterogeneous, the DSs are composed of a set of power invoices belonged to several users. Each power invoice is at the same time consisting of a set of month invoices belonged to a specific user. Each month invoice has information about: customer's contact details, invoice issue date, customer ID (identification number that identifies a user through a supply point at a specific address), invoice number, services hired, and consumption details. There is three types of power: active, reactive, and apparent power. The users contract a specific amount of each kind of power, and then, they consume another amount (more power or less power). Therefore, the consumption details are determined by the hired and the consumed power.

The objective is to determine the minimum cost with regards to the consumption of each customer being necessary to create thousands of optimisation problems, one of each customer. For example, which is the minimum cost for a customer that consumed a specific amount of power during a year.

6.2. Specification of DM, DMap, COP, operators, and system configuration

The DS is formed of IN and OT attributes, according to the mapping with the DM. More attributes can also participate in the DS, but in the used example, all attributes are IN or OT. Following is included the description of each DS attribute:

- **Specification of IN and OT of the DM and the** $\text{DMap}_{\text{DS}\rightarrow\text{DM}}$**:**
  - **IN attributes of the DS:**
    - *CustomerID*: customer identification that uniquely defines a customer. It is the ID of the person.
    - *PowerConsumption*: power consumption over a period such as twelve months or more. Each period is defined by a triple such as $\langle p1, p2, p3\rangle$, for instance $\{8.0, 8.0, 10.0\}$ that represents the power consumption in *kwh*.
    - *Period*: the period in which each record of the DS was captured, such 365 as the number of days of information stored.
    - *TariffHired*: rates hired by the customer.
  - **OT attributes of the DS:**
    - *PostalCode*: location for the customer service, for instance, zip code.
    - *CustomerContactDetails*: extra contact information about the customer.
- **COP**: the constraints of the problem are imposed by the local regulation [20] depending on the rate and power consumption hired by a customer. To illustrate the case, a piece of the code of constraints is included in Fig. 11.
- **Specification of the IN and OUT of the DM and the** $\text{DMap}_{DM\rightarrow\text{COP}}$:
  - **IN attributes of the Data Model related to the COP:**
    - *CustomerID* of the DM is related to the variable *customer* of the COP.
    - *PowerConsumption* of the DM is related to the COP variable *currentConsumption*.
    - *Period* of the DM is related to the variable *comsumption. days* of the COP.
    - *TariffHired* of the DM is related to the variable *Price*.
- **Specification of the OUT attributes of the DM related to the COP:**
  - *EstimatedCost*: optimised cost, this data should improve the real price. This attribute is mapping to the variable *TotalPrice* of the COP.
  - *EstimatedPower*: the optimised power which improves the real power consumption. This is given by triple $\langle p1, p2, p3\rangle$ similar

[2] DC/OS by D2iQ: https://dcos.io/.

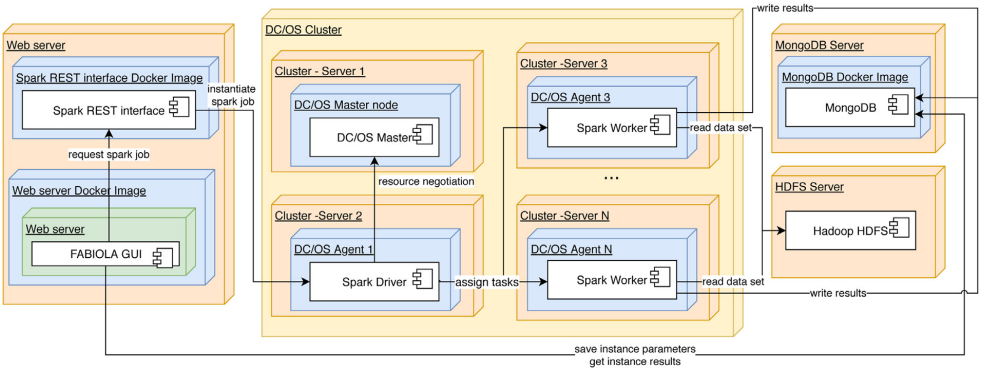


Fig. 10. FABIOLA cluster architecture.

```
Variables and Domains: {
   Var custormer = String;
   Var CurrentConsumption[12][3] = IN.powerconsumption;
   Var Days[12][3] = IN.consumptions.days
   Var Price[3] = IN.TariffHired
   Var TotalCost = intVar(0,MAX_INT_BOUND);
   Var HiredPower[3] = intVar(0,MAX_INT_BOUND);
   Var PowerInvoice[12][3] = intVar(0,MAX_INT_BOUND);
   Var PowerTerm[12][3] = intVar(0,MAX_INT_BOUND);
   Var TotalPrice = intVar(0,MAX_INT_BOUND)}
Constraints: {
   forEach(i <- 0 until 12, j <- 0 until 3) {
     if(HiredPower[j] * 85 > CurrentConsumption[i][j] * 100)
     then(100 * PowerInvoice[i][j] - 85 * HiredPower[j] = 0)

     if(HiredPower[j] * 105 < CurrentConsumption[i][j] * 100)
     then(100 * PowerInvoice[i][j] 210 * HiredPower[j] = 300 * CurrentConsumption[i][j])

     if(HiredPower[j] * 85 < CurrentConsumption[i][j] * 100 AND HiredPower[j] * 105 >=
        CurrentConsumption[i][j]* 100)
     then(100 * PowerInvoice[i][j] = 100 * CurrentConsumption[i][j])

     PowerTerm[i][j] = PowerInvoice[i][j] * Price[j] * Days[i][j]
   }
   forEach(i <- 0 until 12) {
     TP[i] = Sum(PowerTerm[i])
   }
}

Objective Function: {
   TotalPrice = Sum(TP)
   MINIMIZE(TotalPrice)
```

Fig. 11. COP for the optimisation of customer cost.

to the *PowerConsumption*. These attributes are mapping to the variable *HiredPower* of the COP.

- **System Configuration:** the configuration of the system will depend on the characteristics of the problem, but some configurations must the described mandatory in every problem:
  - –**Node Master cores**: one driver is needed at least.
  - –**Node Master memory**: at least one gigabyte for the master node is needed.
  - –**Node Executor core**: the number of executors is determined automatically regarding the size of the job, the available resources, and the load of the cluster. However, one executor at least will be used.
  - –**Nodes Executor memory**: the memory should be tuned in function of the load of the problem, in this particular case with four gigabytes at least per executor is enough.
- **Application of Operators:** the use of operators can help to extract the needed information, for instance, some queries for the example are:
  - –*Which is the region where the power consumption is minimum or less than 400 kWh per month?* This question implies at an *aggregation* of solutions per regions and the application of a MapReduce of solutions to determine the minimum. Moreover, the solutions obtained probably will require some *projections* to represent the results as customer required.
  - –*Which is the average of power consumption of customers with an A3.0 hired tariff per regions?* This question implies a *selection* to manage only the tariff A3.0, a *projection* of per hired tariff afterwards an *aggregation* which enables the average consumption per customer.
  - –*Which is the average of optimisation in a specific region?* First, the optimisation is determined per customer without an oper-

Table 3
Data set information.

Data set	Size (MB)	Number of users
DS1	2058.98	530,504
DS2	441.2	37,279
DS3	2902.29	373,701

ator, afterwards an *aggregation* of those results per region is needed to determine the average of optimisation per region.

6.3. Empirical evaluation and results

In order to evaluate our proposal, several benchmarks have been carried out. The benchmarks have been run over three real DSs (DS1, DS2, DS3) of power providers of Spain. Each DS consists of information respect to the power consumption of customer for several years with the data presented in the previous subsection. The information about the size and the number of customers that includes each DS are given in Table 3.

We have developed two benchmarks whose objectives are:

- *Benchmark I*: *Which is the optimised power consumption to improve (i.e., reduce) the cost of all customers?*
- *Benchmark II*: *Which is the best profit regarding the optimised cost?*

Thus, *Benchmark I* pursues the optimisation of the power consumption of all customers to reduce their costs, and *Benchmark II* pursues the determination of the best (i.e., minimum cost) profit in terms of improvement (i.e., maximum) of the estimated cost. The *Benchmark I* is based on the idea of Computation of independent COPDD (cf., Section 4.1) since the COPs are independent and any operators are required to combine solutions, while *Benchmark II* is based on the Computation COPDD with domain reduction (cf., Section 4.2) since operators of *selection, aggregation and projection* could be required.

6.3.1. Benchmark I: computation of independent COPDDs

Regarding the computational impact, this benchmark aims to demonstrate the impact of the execution of COPs in stand-alone (i.e., sequentially in just one node) and using FABIOLA (i.e., parallelised). The COPs to be enacted in both cases are the same, therefore, they are comparable and compatible. This comparison is performed by evaluating the scalability as the number of COPs increases by means of asymptotic analysis.

For this study, DS1, DS2 and DS3 are employed. For each one, an analysis of the scalability of our proposal is performed. This analysis is carried out by progressively increasing the size of each DS. For instance, DS1 is scaled five times. Then, five executions are performed for each one, and the average Elapsed Real Time (ERT)[3] is calculated. This will enable us to analyse the temporal complexity from the user's point of view as the number of problems increases. This test is performed in sequential as well as in parallel.

Regarding the configuration of the experiment, all the resources of the cluster previously described are employed. It means that there are four Spark workers for solving the COPs in parallel, each one provisioned with four cores and 16 gigabytes of main memory. On the other hand, the *timeout* is fixed to 5 s.

Although several metrics have been captured in the test, we have them grouped in three types of metrics: (i) problem modelling; (ii) execution performance (in time and memory), and; (iii) impact of *timeout*.

[3] The ERT is the time from the start of a program to its end.

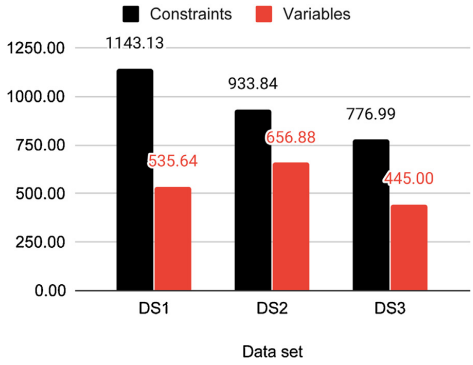


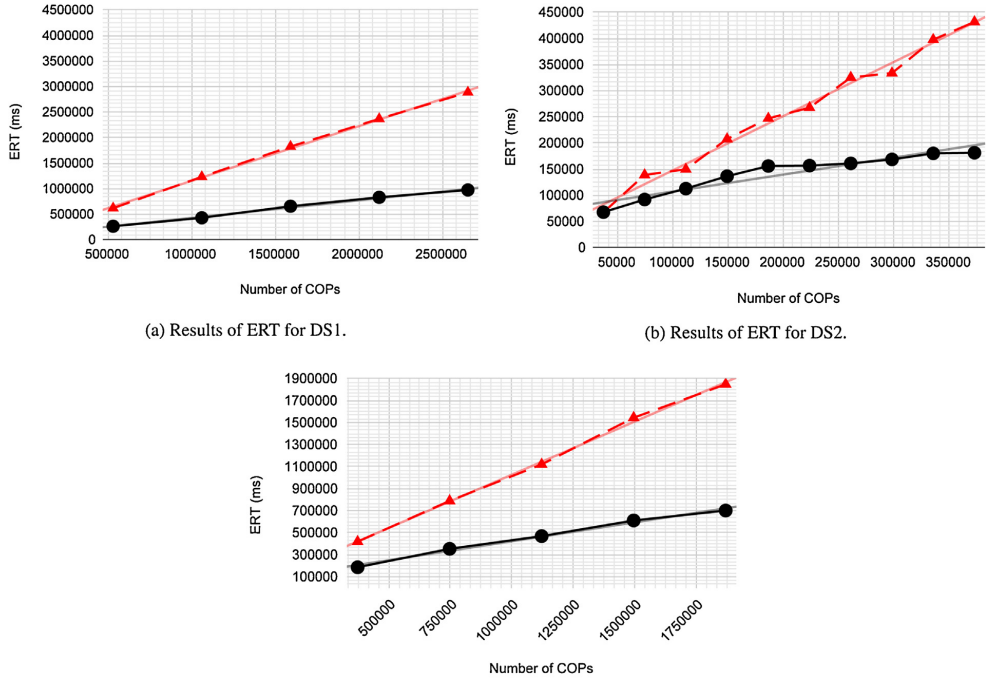
Fig. 12. Number of constraints and variables.

Regarding **problem modelling** metrics, for each DS, the number of constraints and variables resulting from the COP model is quantified. The number of constraints and variables depends on the attributes of the DS and the solver. The solver will require instantiating internal constraints and variables. Note that the average number of constraints and variables is not affected when proportionally scaling the DS, since the proportion will always be the same. Moreover, it is not affected by whether the execution is sequential or parallel. It only depends on the DS, and hence, these results have been depicted by DS. These are depicted in Fig. 12.

From the results in the figure, it is possible to conclude that the complexity of DS3 might be minor than the other two DSs, since it has the smallest number of variables. On the other hand, it is impossible to ensure that the complexity of this DS is less since there are other factors that can condition the complexity, for instance, constraints can affect the complexity of the COPs. The more constraint, the more the domain is limited, hence, a greater number of constraints would imply (in theory) less complexity. Albeit to ensure that, every single constraint would have to be analysed in order to see how it affects the domain of the variables and it is not a trivial task.

Regarding the **execution performance** metrics, asymptotic analysis is carried out. Fig. 13 depicts the results of these tests for the three DSs, showing how the ERT varies as the number of COPs increases. While DS1 and DS3 have been scaled five times, DS2 has been scaled 10 times. It is due to the fact that DS2 is smaller than DS1 and DS3, and scaling it just five times is not enough to compare it to the other two DSs. In addition, the behaviour in the first five tests in parallel differs from the behaviour in the last five. Note that the ERT tends to converge after the fifth point (i.e., after scaling the DS five times). On the other hand, the execution of DS2 without scaling it (cf., first black point in the chart (b)) yields better results sequentially than in parallel. From that point, the execution in parallel improves as the size of the DS increases. It is due to the fact that the size of the DS is very small, thereby it is not worth executing it in parallel since to distribute the COPs among the cluster nodes involves a high cost compared to the resolution time.

As can be appreciated, the sequential ERT is worse than the parallel. In addition, the executions in parallel tend to behave better as the size of the problem (i.e., the number of COPs) increases. In order to analyse the limiting behaviour of our proposal (i.e., an asymptotic analysis), the trend-lines corresponding to each set of executions has been calculated in Table 4. The equations of the trend-lines are shown. In these equations, y stands for the dependent variable (i.e., the ERT), while x stands for the independent variable (i.e., the number of COPs to solve).



(a) Results of ERT for DS1.

(b) Results of ERT for DS2.

(c) Results of ERT for DS3.

Fig. 13. Each point and triangle are the ERT average for five executions. Red and black line represent the sequential an parallel execution respectively.

Table 4
Trend-line equations for the execution performance of sequential and parallel solutions per DS.

Data set	Sequential	Parallel
DS1	$y = 1.07x + 84245$	$y = 0.34x + 84569$
DS2	$y = 1.04x + 43887$	$y = 0.32x + 74748$
DS3	$y = 0.97x + 59582$	$y = 0.34x + 77845$

The equation slopes are similar for all the DSs. It means that the differences between the complexities of the three DSs do not highly impact the performance. As can be observed, the slope is approximately 1 for the sequential executions. It implies a linear complexity (i.e., $\mathcal{O}(n)$). Regarding the parallel execution, the slope is approximately 3 times less than the linear slope. Hence, the complexity is sub-linear (i.e., $\mathcal{O}(\frac{n}{3})$). From these results, it is possible to conclude that the execution of COPs in parallel improves the sequential execution. Remark that, as observed with DS2, the parallel execution is not always worth. There is an intersection point from which the parallel execution improves the sequential. For DS1, this point is reached at 444; for DS2, the intersection is reached at 43, 102, and; for DS3, it is reached at 29, 314. For a number of COPs less that those values, the parallel distribution is not worth.

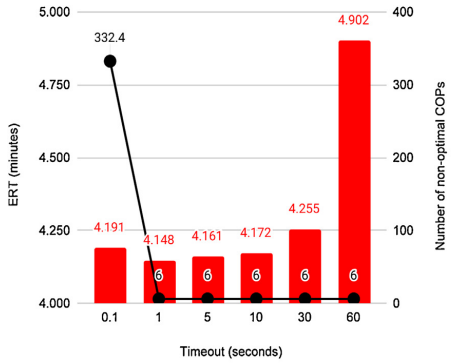
Regarding the **impact of the timeout**, six different values for the *timeout* have been studied on the DSs: 0.1, 1, 5, 10, 30, and 60 s. Thereby, for each *timeout*, five executions are performed, and the average ERT and number of non-optimal COPs are calculated. While the ERT is expected to increase as the *timeout* increases, the number of non-optimal COPs is expected to decrease.

Fig. 14 shows the results for this study. Regarding the ERT, it increases when as the *timeout* increases except for small *timeout* values. It is due to the fact that small *timeout* values do not affect the execution, especially when the number of non-optimal COPs is small compared to the size of the dataset. On the other hand, larger *timeout* values do cause an impact on the ERT. In this case, the few COPs that have not been optimally solved (approx. 6 for DS1 and DS3) are creating bottlenecks.

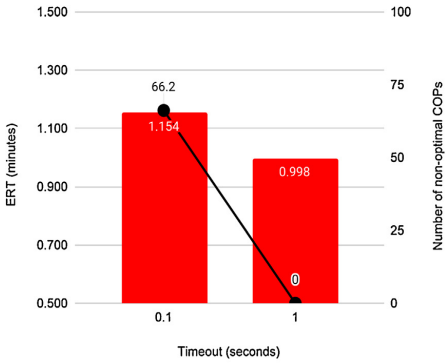
Remark that for DS2, all COPs were optimally solved with a *timeout* of 1 second. For this reason, other timeout values have not been checked since the behaviour would be similar. Contrary to the expected, the ERT when the *timeout* is 1 second. The previous reasoning applies here. These values of timeout are so small that COPs will not behave as bottlenecks.

These results support the use of five seconds as *timeout* for the tests that have been carried out in this section since it offers a good balance between the ERT and the number of COPs that are optimally solved.

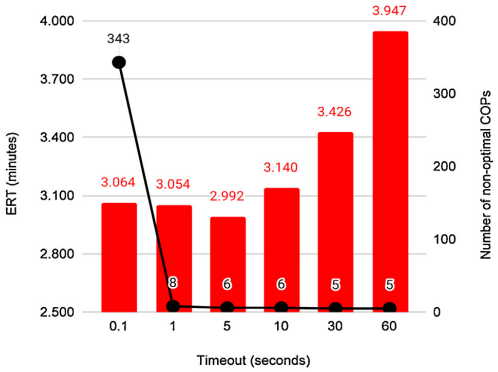
Regarding the **memory performance**, it depends on how the program is planned and the distribution of the workload among the nodes. In this case, the program is composed of two phases. While the first one consists of reading the data set from the data sources, the second phase is based on building and solving the COPs. Then, the aggregated amount of memory approximate linearly to the size of the data set independently whether or not the execution is sequential or parallel as shown in Fig. 15. This phenomenon is due to the initial read and distribution, the data to be processed is not distributed among the nodes of the cluster. Remark that all of them present an upper linear slope, the differences among them are due to the differences in the average size per COP in each DS. For instance, the average COP size is 1.2×10^{-5} KB for DS2, 7.8×10^{-6} KB for DS3, and 3.9×10^{-6} KB for DS1.



(a) Timeout tests results for DS1.



(b) Timeout tests results for DS2.



(c) Timeout tests results for DS3.

Fig. 14. Behaviour of the ERT and number of non-optimal COPs as the timeout increases. Red columns represent the average ERT of five executions, and black lines represent the average number of non-optimal COPs of five executions.

6.3.2. Benchmark II: computation COPDDs with domain reductions

All the metrics in the previous section can help to understand the problem in the form of the COPs and the impact of parallelisation in the resolution of COPs. In the Benchmark II, we want to demonstrate other aspects related to the application of operators and their impact in the time of solution.

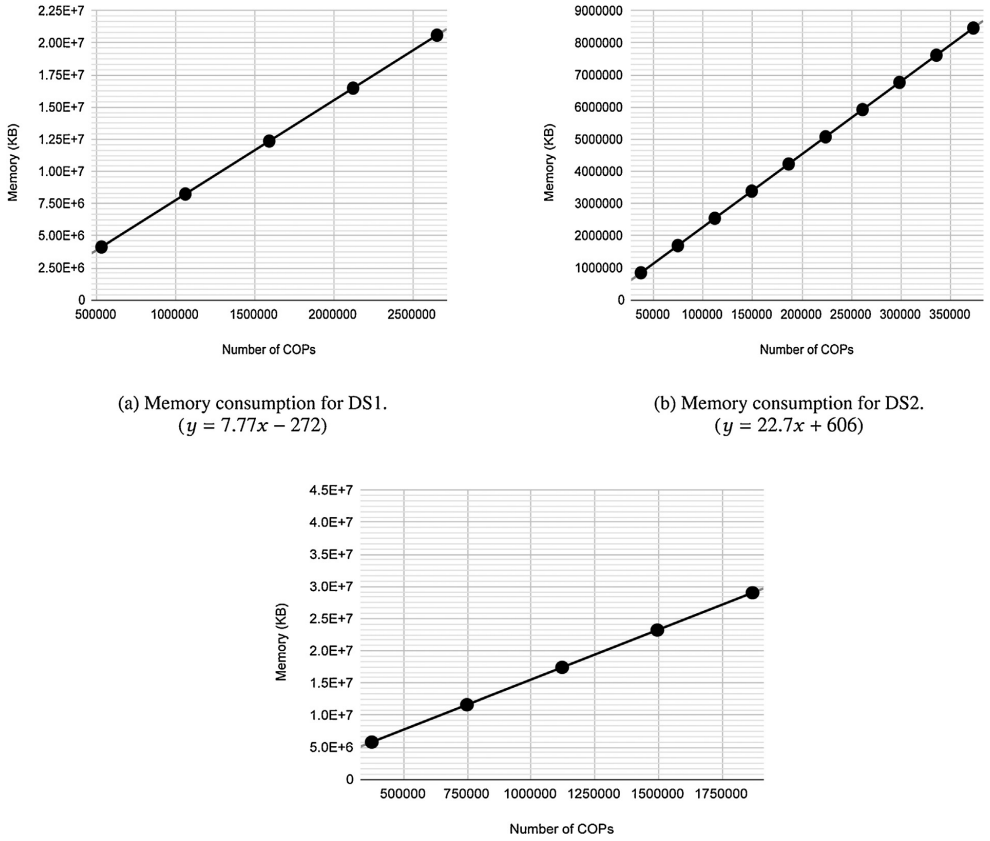
This benchmark pursues the determination of the best profit (i.e., minimum cost) in terms of improvement (cf., maximum) of the estimated cost. First, an *aggregation operator* is applied to range the customer from low to the high hired tariff. Afterwards, the customers are split into *workgroups*. The *reduction operation* is applied since the minimum cost is pursued the previously estimated cost is used to feed the unsolved COPs into the same *workgroup*. When all the customers have to be processed the minimum of each *workgroup* have to be combined (cf. MapReduce) in order to determine the global minimum.

In terms of the COP resolution, the objective is the reduction of the domain of the variables, by introducing an upper bound to the objective variable.

This experiment is similar to the asymptotic analysis performed in Benchmark I, but only parallel executions have been carried out. Fig. 16 depicts the results of this study, and Table 5 shows the trend-line equations for each DS. As can be seen, the aggregation operator not only outperforms the execution without such operator, but worsens the scalability (cf., note that the slopes are 0.54, 0.57 and 0.76 for DS1, DS2 and DS3, respectively). Nonetheless, by comparing the COP resolution stages of the execution, it can be checked that the reduction of the domain improves both the run-time and the scalability, except for the two first executions of DS2. It might be due to the fact that those points represent small data sets and the improvement is not noticeable by limiting the domain when running in parallel. Regarding the scalability, for DS3 the slope is slightly higher when applying reductions, which might

Table 5
Trend-line equations for the execution performance per DS.

Data set	Aggregation and COP resolution	COP resolution after reduction	COP resolution without reduction
DS1	$y = 0.54x - 23300$	$y = 0.28x + 141$	$y = 0.31x + 64862$
DS2	$y = 0.57 + 101583$	$y = 0.14x + 56593$	$y = 0.24x + 56840$
DS3	$y = 0.76x + 9763$	$y = 0.27x + 750$	$y = 0.26x + 65171$



(a) Memory consumption for DS1. ($y = 7.77x - 272$)

(b) Memory consumption for DS2. ($y = 22.7x + 606$)

(c) Memory consumption for DS3. ($y = 15.5x - 91.8$).

Fig. 15. Memory consumption as the number of COPs scale progressively. Each point represents the average of the total of five executions (Kbytes).

be due to the fact that the domain reduction for this data set is not very significant.

To sum up, the application of aggregation operators does not improve the execution performance in terms of time, but it might improve the COP resolution stage of the execution depending on the nature of the data. However, this improvement might not be significant since the aggregation stage takes from the 40% to the 50% of the execution time, and also scales worse as the number of COPs increases.

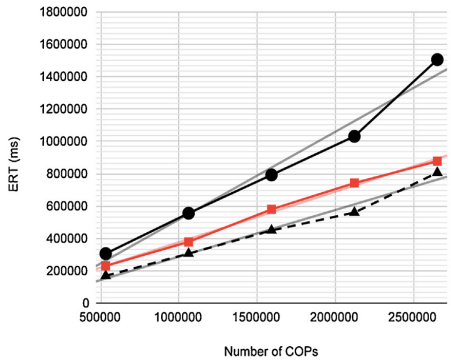
Respect to the **percentage of optimised** COPs, when applying this aggregation operator there is a high amount of COPs which are not optimally solved. Note that reducing the COP domains might lead to unfeasible problems. Hence, the proportion of solved COPs respect to the all possible COPs are 12.11% for DS1; 12.38% for DS2, and 24.9% for DS3.

Regarding the **memory performance**, in this case the program is composed of three stages. The read phase from the data sources and the COP resolution phase are the similar as Benchmark I, but between those stages another phase is included which consists of aggregating the data. This operation implies an extra memory consumption, since the records from the data set must be redistributed among the nodes of the cluster. Fig. 17 shows these results. Since applying aggregation operators require more memory consumption, these outperform the previous results. It can be checked both in the values of memory consumed and in the slope of the trend-lines, which are higher than in the previous case, implying a worsening in the scalability.

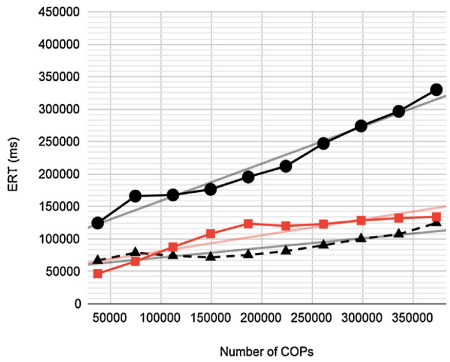
7. Related work

Big Data faces up new challenges [25,31] in how to carry out optimisation problems with heterogeneous, incomplete, and uncertain data, besides, to immediate responses for some types of questions.

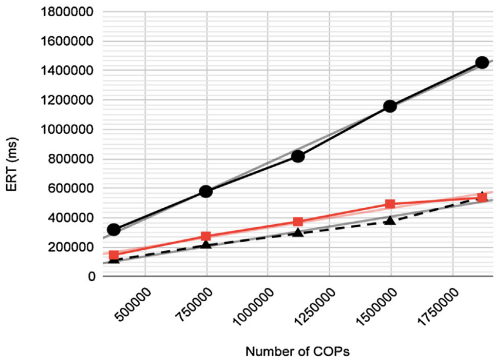
The Apache Hadoop project [3] actively supports multiple projects to extend Hadoop's capabilities and make it easier to use. There are several excellent projects to helping in the creation of development tools as well as for managing Hadoop data flow and processing. Many commercial third-party solutions build on their developed technologies within the Apache Hadoop ecosystem. Spark [49], Pig [32], and Hive [45] are three of the best-known Apache Hadoop projects. All of them are used to create applications to process Hadoop data. While there are a lot of articles and discussions about which is the best one, in practice, many organisations



(a) Results of ERT for DS1.



(b) Results of ERT for DS2.



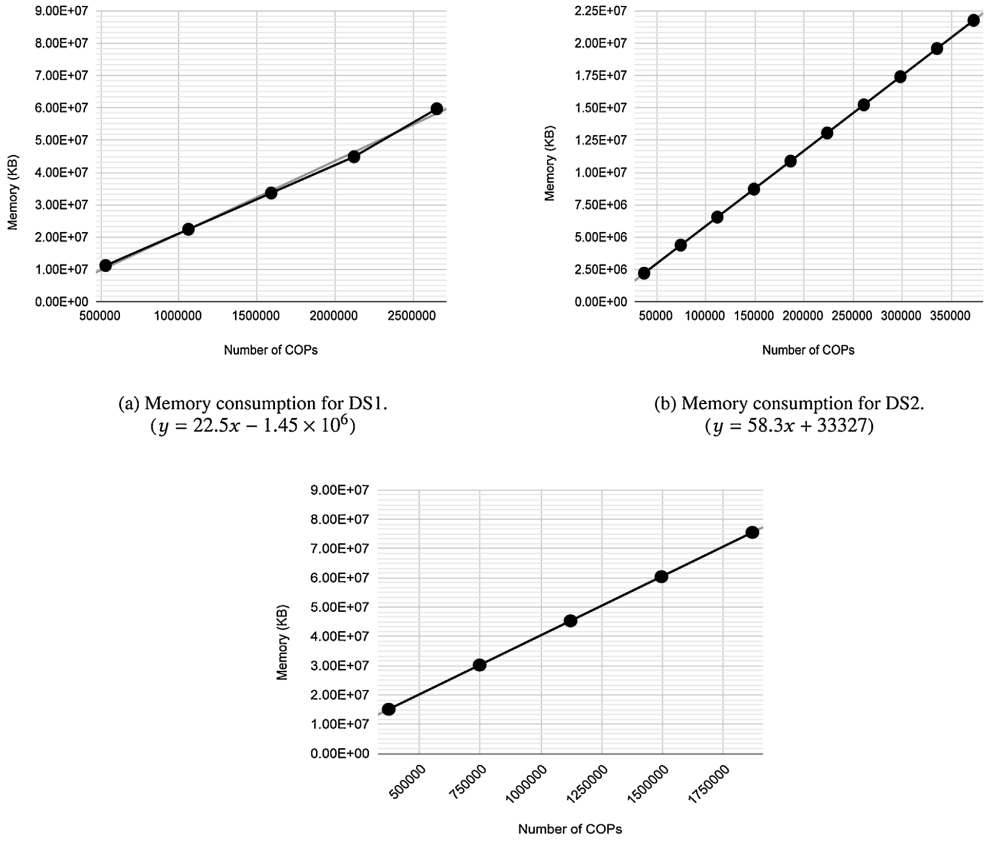
(c) Results of ERT for DS3.

Fig. 16. Each point, triangle and square is the Elapsed Real Time average for five executions. Black continuous line represents the overall Elapsed Real Time of the execution including aggregations and COP resolutions; black pointed line represents the Elapsed Real Time of the COP resolutions with domain reduction, and red line represents the Elapsed Real Time of the COP resolution without domain reduction.

use various of them since each one is optimised for specific functionality. Although FABIOLA is not a new Big Data solution, it aims to be part of the Hadoop ecosystem. FABIOLA provides the necessary components to drive the solution of COPs with distributed data on a Hadoop-based architecture.

In [54], Zhu et al. present various proposals that solve some of the challenges for optimisation problems in distributed information systems. The proposals focus on the optimisation of systems performance and three of them combine optimisation problems in big data environments. First, Zeng et al. [50] emphasised minimising the cost of renting cloud resources for executing MapReduce applications in public clouds, and propose a greedy-based scheduling algorithm to tackle this issue. Secondly, Zhou et al. [53] efficiently distribute data on different devices by considering the asymmetric characteristics among devices and data. To do that, they propose a preference model to quantitatively weight the storage performance imbalance when data are distributed on different devices. Finally, Zhao et al. [52] propose an algorithm to solve the data allocations under multiple constraints in polynomial time. However, these proposals are focused on performance optimisation and not on optimising the results obtained. On the other hand, Chen et al. [8] present a survey where challenges and opportunities of Big Data in data-intensive systems are identified. The authors identified the opportunity to improve decision-making focused on the customer, for instance, by developing customised products. In order to do that, the authors identified the techniques and technologies of Big Data. The authors highlight the need for the application of optimisation techniques to efficiently process a large volume of data within limited run times. Several optimisation techniques are mentioned (cf., Mathematical tools) however they missed out CP as an optimisation technique.

CP presents a challenge in the scalability by solving some hard problems. However, CP has been successfully applied in different domains for solving optimisation problems, such as scheduling and planning. Although there exist several CP tools, such as IBM CPLEX Optimisation [23] and ChocoSolver [37] and solutions that create Constraint Problems according to the constraints extracted from databases [5,18,19,17,38], none of them provides an integrated solution for a Big Data solution. Big Data provides to CP a new perspective concerning the size and volume of data, and it is an excellent opportunity to exploit its possibilities to gain efficiency and optimisation in operational processes [34]. Additionally, Big Data tackles new challenges [16] dealing with automation of decision-making that involves several (millions) decision variables



(a) Memory consumption for DS1. ($y = 22.5x - 1.45 \times 10^6$)

(b) Memory consumption for DS2. ($y = 58.3x + 33327$)

(c) Memory consumption for DS3. ($y = 40.4x + 4898$).

Fig. 17. Memory consumption as the number of COPs scale progressively. Each point represents the average of the total of five executions (Kbytes).

in optimisation of resource consumption, sustainability services, and finance. Nevertheless, the optimisation problems in CP need more flexibility and adaptability, since the exploration of heterogeneous, enormous and dynamic generation of data requires a quick adaptation of optimisation problem to more holistic solutions.

DCOPs have been widely tackled in the literature [15,26,29,48,27,47]. Due to the complexity of the problems (NP-complete, [7,4]) most of the efforts have been done in a theoretical way with the proposition of models, algorithms, and strategies to solve this type of problems. Only ad-hoc applications [29], heuristic approaches [14,13] or simulators [15] have been developed for the community to solve those type of problems. The main challenges in the DCOP solvers are based on overcoming the massive use of memory needed to run the algorithms and to manage the highest number of messages exchanged between nodes. Currently, none approach of DCOPs applied to Big Data has been identified in the literature.

To the best of our knowledge, FABIOLA is the first approach that applied Big Data in CP, although various have been the examples where the necessity to optimise problems with Big Data has been detected [51,24]. There are seminal works related to the application of Big Data in CP. Sunny-CP [1,2] is a portfolio of CP-solvers which can exploit the multicore capabilities of the machines to solve CSPs and COPs. However, this approach lacks applicability in the context of Big Data with multiple data sources with a large-scale amount of data. Cao et al. [6] study how to improve the solution of a large-scale Integer Linear Programming (ILP) problems employing Big Data architecture. ILP is a discrete approach compared with CP. The authors applied ILP to solve traffic flow management. In [28], the authors propose the application of a CP-based scheduler for a Big Data architecture. On the other hand, there is an initiative to create a new language to adapt CP languages for Big Data applications [40], it is currently a very undeveloped approach with no continued development.

Other approaches to Big Data and Optimisation techniques applied to real cases have been studied in [12]. For instance, the maritime logistics problem is modeled as a constraint problem and optimisation techniques have been applied although most of the approaches are focused on machine learning or logic programming techniques.

8. Conclusions and future work

Optimisation problems are found in several real-life scenarios. These optimisation problems become an extraordinary problem when the data involved are in a Big Data environment, which implies a massive quantity of information, that is also distributed and heterogeneous. FABIOLA has been developed to support the definition and resolution of COPs within Big Data scenarios. FABIOLA isolates the resolution of optimisation problems from where the data are and how the optimal outputs are found. Thanks to this isolation, the resolution of optimisation problems in Big Data environments has never been so simple as with FABIOLA. Internally, FABIOLA has been provided with a set of operators to query the optimal information obtained. In order to reduce the computational time and resources needed to solve the optimisation problems, the operators are applied to enrich the capabilities of the resolution of COPs by using the MapReduce operations. Besides, our solution facilitates the COPs modelling through a user interface, FABIOLA-UI, that guides the final users through the application. FABIOLA has been evaluated in real scenarios where the necessity of improving the billing of thousands of customers by using COPs is essential. FABIOLA demonstrated to be a scalable and adaptable solution that improves trivial and stand-alone solutions. However, the most important results are the incredible improvement regarding the performance of execution and resource consumption in optimisation problems. The resolution of optimisation problems in Big Data environments has never been as simple to model and fast to solve as with FABIOLA.

As future work, it would be interesting to extend FABIOLA to support the adaptation of various constraint solvers in the architecture. On the other hand, FABIOLA can be extended in many directions, for instance, related to the inclusion of new operators to optimise the queries.

The possibilities of extends FABIOLA in many directions related to the operators and other scenarios such as distributed optimisation problems are desirable.

Authors' contribution

All the authors are responsible for the concept of the paper, the results presented and the writing. All the authors have approved the final content of the manuscript. No potential conflict of interest was reported by the authors.

Conflict of interest

All the authors are responsible for the concept of the paper, the results presented and the writing. All the authors have approved the final content of the manuscript. No potential conflict of interest was reported by the authors.

Acknowledgements

This research was partially supported by Ministry of Science and Technology of Spain with projects ECLIPSE (RTI2018-094283-B-C33) and by Junta de Andalucía with METAMORFOSIS projects; the European Regional Development Fund (ERDF/FEDER); and by the University of Seville with *VI Plan Propio de Investigación y Transferencia* (VI PPIT-US).

References

[1] R. Amadini, M. Gabbrielli, J. Mauro, A multicore tool for constraint solving, Buenos Aires, Argentina, July 25–31, in: Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015, 2015, 2015, pp. 232–238.

[2] R. Amadini, M. Gabbrielli, J. Mauro, SUNNY-CP and the minizinc challenge, TPLP 18 (2018) 81–96.

[3] Apache, Apache hadoop, 2018, Online. http://wiki.apache.org/hadoop (Accessed 18 May 2018).

[4] C. Bessiere, I. Brito, P. Gutierrez, P. Meseguer, Global constraints in distributed constraint satisfaction and optimization, Comput. J. 6 (2013).

[5] D. Borrego, R. Eshuis, M.T. Gómez-López, R. M. Gasca, Diagnosing correctness of semantic workflow models, Data Knowl. Eng. 87 (2013) 167–184, http://dx.doi.org/10.1016/j.datak.2013.04.008.

[6] Y. Cao, D. Sun, Large-Scale and Big Optimization Based on Hadoop, Springer International Publishing, Cham, 2016, pp. 375–389.

[7] P. Cheeseman, B. Kanefsky, W.M. Taylor, Where the really hard problems are, in: Proceedings of the 12th International Joint Conference on Artificial Intelligence - vol. 1, Morgan Kaufmann Publishers Inc, San Francisco, CA, USA, 1991, pp. 331–337 http://dl.acm.org/citation.cfm?id=1631171.1631221.

[8] C.P. Chen, C.Y. Zhang, Data-intensive applications, challenges, techniques and technologies: a survey on big data, Inform. Sci. 275 (2014) 314–347, http://dx.doi.org/10.1016/j.ins.2014.01.015.

[9] G. Cornuejols, M. Trick, Quantitative Methods for the Management Sciences 45-760. Course Notes, 1998.

[10] J. Dean, S. Ghemawat, Mapreduce: simplified data processing on large clusters, Commun. ACM 51 (2008) 107–113, http://dx.doi.org/10.1145/1327452.1327492.

[11] K.R. Duffy, C. Bordenave, D.J. Leith, Decentralized constraint satisfaction, IEEE/ACM Trans. Netw. 21 (2013) 1298–1308, http://dx.doi.org/10.1109/TNET.2012.2222923.

[12] A. Emrouznejad, Big Data Optimization: Recent Developments and Challenges, Springer, 2016.

[13] D. Fernández-Cerero, A. Fernández-Montes, J.A. Ortega, Energy policies for data-center monolithic schedulers, Expert Syst. Appl. 110 (2018) 170–181.

[14] D. Fernández-Cerero, Á.J. Varela-Vaca, A. Fernández-Montes, M.T. Gómez-López, J.A. Alvárez-Bermejo, Measuring data-centre workflows complexity through process mining: the Google cluster case, J. Supercomput. (2019), http://dx.doi.org/10.1007/s11227-019-02996-2.

[15] F. Fioretto, E. Pontelli, W. Yeoh, Distributed constraint optimization problems and applications: a survey, J. Artif. Intell. Res. 61 (2018) 623–698, http://dx.doi.org/10.1613/jair.5565.

[16] E.C. Freuder, B. O'Sullivan, Grand challenges for constraint programming, Constraints 19 (2014) 150–162, http://dx.doi.org/10.1007/s10601-013-9155-1.

[17] M.T. Gómez-López, R. Ceballos, R.M. Gasca, C.D. Valle, Developing a labelled object-relational constraint database architecture for the projection operator, Data Knowl. Eng. 68 (2009) 146–172, http://dx.doi.org/10.1016/j.datak.2008.09.002.

[18] M.T. Gómez-López, R.M. Gasca, Using constraint programming in selection operators for constraint databases, Expert Syst. Appl. 41 (2014) 6773–6785, http://dx.doi.org/10.1016/j.eswa.2014.04.047.

[19] M.T. Gómez-López, R.M. Gasca, Object relational constraint databases for GIS, Encyclopedia of GIS (2017) 1449–1457, http://dx.doi.org/10.1007/978-3-319-17885-1_1598.

[20] Red Eléctrica Group, 'red eléctrica espa na', 2018, Available at http://www.ree.es/es.

[21] P. Gutiérrez Faxas, Distributed Constraint Optimization Related With Soft Arc Consistency, Universidad Autónoma de Barcelona. Departamento Ciencias de la Computación, 2013, Ph.D. thesis.

[22] J.M. Hellerstein, J. Heer, S. Kandel, Self-service data preparation: research to practice, IEEE Data Eng. Bull. 41 (2018) 23–34.

[23] IBM, Ibm-ilog Cplex Optimization. https://www.ibm.com/products/ilog-cplex-optimization-studio (Accessed January 2018).

[24] K. Kolomvatsos, C. Anagnostopoulos, S. Hadjiefthymiades, An efficient time optimized scheme for progressive analytics in big data, Big Data Res. 2 (2015) 155–165, http://dx.doi.org/10.1016/j.bdr.2015.02.001.

[25] A. Labrinidis, H.V. Jagadish, Challenges and opportunities with big data, Proc. VLDB Endow. 5 (2012) 2032–2033, http://dx.doi.org/10.14778/2367502.2367572.

[26] T. Le, T.C. Son, E. Pontelli, W. Yeoh, Solving Distributed Constraint Optimization Problems Using Logic Programming, 2017 arXiv:1705.03916.

[27] A.R. Leite, F. Enembreck, J.P.A. Barthas, Distributed constraint optimization problems: review and perspectives, Expert Syst. Appl. 41 (2014) 5139–5157, http://dx.doi.org/10.1016/j.eswa.2014.02.039.

[28] N. Lim, S. Majumdar, P. Ashwood-Smith, A constraint programming based Hadoop scheduler for handling mapreduce jobs with deadlines on clouds, in: Proceedings of the 6th ACM/SPEC International Conference on Performance Engineering, ACM, New York, NY, USA, 2015, pp. 111–122, http://dx.doi.org/10.1145/2668930.2688058.

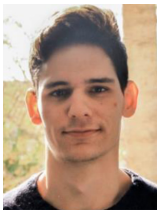
[29] D.K. Molzahn, F. Dörfler, H. Sandberg, S.H. Low, S. Chakrabarti, R. Baldick, J. Lavaei, A survey of distributed optimization and control algorithms for electric power systems, IEEE Trans. Smart Grid 8 (2017) 2941–2962.

[30] Mongo, Mongodb, 2018, Available at https://www.mongodb.com/.

[31] T. Nasser, R. Tariq, Big data challenges, Comput. Eng. Inform. Technol. 4 (2015) 1–10, http://dx.doi.org/10.4172/2324-9307.1000133.

[32] C. Olston, B. Reed, A. Silberstein, U. Srivastava, Automatic optimization of parallel dataflow programs, USENIX 2008 Annual Technical Conference, USENIX Association, Berkeley, CA, USA (2008) 267–273 http://dl.acm.org/citation.cfm?id=1404014.1404035.

[33] OMG, The OMG Data-Distribution Service for Real-Time Systems, 2018, Online. http://portals.omg.org/dds/ (Accessed 18 January 2018).
[34] B. O'Sullivan, Opportunities and challenges for constraint programming, in: Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, AAAI Press, 2012, pp. 2148–2152 http://dl.acm.org/citation.cfm?id=2900929.2901033.
[35] L. Parody, Á.J. Varela-Vaca, M.T. Gómez-López, R.M. Gasca, FABIOLA: Defining the Components for Constraint Optimization Problems in Big Data environment, Information System Development - Improving Enterprise Communication, [Proceedings of the 26th International Conference on Information Systems Development, ISD 2017, Larnaca, Cyprus] (2017).
[36] L. Parody, Á.J. Varela Vaca, M.T. Gómez López, R. M. Gasca, Fabiola: Towards the resolution of constraint optimization problems in big data environment, in: Paspallis, N, Raspopoulos M, Barry C, Lang M, Linger H, C. Schneider (Eds.), Advances in Information Systems Development, Springer International Publishing, Cham, 2018, pp. 113–127.
[37] C. Prud'homme, J.G. Fages, X. Lorca, Choco Documentation. TASC – LS2N CNRS UMR 6241, COSLING S.A.S., 2017 http://www.choco-solver.org.
[38] P.Z. Revesz, Introduction to Constraint Databases. Texts in Computer Science, Springer, 2002, http://dx.doi.org/10.1007/b97430.
[39] F. Rossi, P. van Beek, T. Walsh, Handbook of Constraint Programming, Elsevier, 2006.
[40] F. Rossi, V. Saraswat, Constraint Programming Languages for Big Data Applications, 2010 https://github.com/saraswat/C10.
[41] R. Sahal, M.H. Khafagy, F.A. Omara, Exploiting coarse-grained reused-based opportunities in big data multi-query optimization, J. Comput. Sci. 26 (2018) 432–452, http://dx.doi.org/10.1016/j.jocs.2017.05.023.
[42] S. Sai, S. Esakkirajan, Fundamentals of Relational Database Management Systems, vol. 47, 2007, http://dx.doi.org/10.1007/978-3-540-48399-1.
[43] M. Soualhia, F. Khomh, S. Tahar, Task scheduling in big data platforms: a systematic literature review, J. Syst. Softw. 134 (2017) 170–189, http://dx.doi.org/10.1016/j.jss.2017.09.001.
[44] J. Stefanowski, K. Krawiec, R. Wrembel, Exploring complex and big data, Int. J. Appl. Math. Comput. Sci. 27 (2017) 669–679.
[45] A. Thusoo, J.S. Sarma, N. Jain, Z. Shao, P. Chakka, S. Anthony, H. Liu, P. Wyckoff, R. Murthy, Hive: a warehousing solution over a map-reduce framework, Proc. VLDB Endow. 2 (2009) 1626–1629, http://dx.doi.org/10.14778/1687553.1687609.
[46] A. Valencia-Parra, A. Varela-Vaca, M. Gómez-López, P. Ceravolo, Chamaleon: framework to improve data wrangling with complex data, ICIS (2019) 205–207.
[47] T. Yang, X. Yi, J. Wu, Y. Yuan, D. Wu, Z. Meng, Y. Hong, H. Wang, Z. Lin, K.H. Johansson, A survey of distributed optimization, Annu. Rev. Control 47 (2019) 278–305, http://dx.doi.org/10.1016/j.arcontrol.2019.05.006.
[48] W. Yeoh, A. Felner, S. Koenig, Bnb-Adopt: An Asynchronous Branch-and-Bound DCOP Algorithm, 2014 arXiv:1401.3490.
[49] M. Zaharia, R.S. Xin, P. Wendell, T. Das, M. Armbrust, A. Dave, X. Meng, J. Rosen, S. Venkataraman, M.J. Franklin, A. Ghodsi, J. Gonzalez, S. Shenker, I. Stoica, Apache spark: a unified engine for big data processing, Commun. ACM 59 (2016) 56–65, http://dx.doi.org/10.1145/2934664.
[50] X. Zeng, S.K. Garg, Z. Wen, P. Strazdins, A.Y. Zomaya, R. Ranjan, Cost efficient scheduling of mapreduce applications on public clouds, J. Comput. Sci. 26 (2018) 375–388, http://dx.doi.org/10.1016/j.jocs.2017.07.017.
[51] Z. Zhang, K.R. Choo, B.B. Gupta, The convergence of new computing paradigms and big data analytics methodologies for online social networks, J. Comput. Science 26 (2018) 453–455, http://dx.doi.org/10.1016/j.jocs.2018.04.007.
[52] H. Zhao, M. Qiu, M. Chen, K. Gai, Cost-aware optimal data allocations for multiple dimensional heterogeneous memories using dynamic programming in big data, J. Comput. Sci. 26 (2018) 402–408, http://dx.doi.org/10.1016/j.jocs.2016.06.002.
[53] W. Zhou, D. Feng, Z. Tan, Y. Zheng, Improving big data storage performance in hybrid environment, J. Comput. Sci. 26 (2018) 409–418, http://dx.doi.org/10.1016/j.jocs.2017.01.003.
[54] X. Zhu, L.T. Yang, H. Jiang, P. Thulasiraman, B.D. Martino, Optimization in distributed information systems, J. Comput. Sci. 26 (2018) 305–306, http://dx.doi.org/10.1016/j.jocs.2018.04.020.



Álvaro Valencia-Parra is a computer science Ph.D. student at the University of Seville. His research areas include the improvement of data transformation and data quality fields in Big Data paradigms. His goal is to improve the way in which final users deal with data preparation and specific scenarios in which configuring a Big Data pipeline might be tricky. For this purpose, he is working in the improvement of the data transformation processes by designing Domain-Specific Languages for very specific purposes, user interfaces, and semi-automatic approaches in order to assist users in these tasks.

Angel J. Varela-Vaca received the B.S. degree in Computer Engineering at the University of Seville (Spain) and graduated in July 2008. M.Sc. on Software Engineering and Technology (2009) and obtained his PhD with honors at the University of Seville (2013). Angel is currently working as Assistant Professor at Languages and System Informatics Department at the Universidad Sevilla and belongs to the Idea Research Group. Angel has and leaded various private projects and participated in several public research projects and he has published several impact papers. He was nominated as a member of Program Committees such as ISD 2016, BPM Workshops 2017, SIMPDA 2018. He has been reviewer for international journals such as Journal of Supercomputing, International Journal of Management Science and Engineering Management Multimedia Tools and Applications, Human-Centric Computational and Information Sciences, Mathematical Methods in Applied Sciences among others.

Luisa Parody studied computer engineering (including a minor in systems engineering) at the Universidad de Sevilla (Spain) and graduated with honours in July 2009. She then earned an M.Sc. degree in software engineering and technology (2010) and obtained her international PhD with honours at the Universidad Sevilla (2014). Since 2018, she has been working as an associate professor in Dto. Método Cuantitativos at the Universidad Loyola. She belongs to the IDEA Research Group and has participated in several private and public research projects and has published several high-impact papers.

María Teresa Gómez-López is a Lecturer at the University of Seville and the head of the IDEA Research Group. Her research areas include Business Processes and Data management, and the things that disturb his thoughts are how to improve the business process models including better decisions and enriching the model with Data Perspectives. She has led several private and public research projects and has published several impact papers, among others in Information and Software Technology, Information Systems, Information & Software Technology, or Data & Knowledge Engineering. She was nominated as a member of Program Committees, such as ER 2018, BPM 2017, ISD 2017, ICIQ 2016, PHM 2016, IAwDQ 2013 or CLEIS 2012. She has been reviewing for international journals, such as International Journal of Data and Information Quality, Journal of Systems and Software, Artificial Intelligence in Medicine; Business & Information Systems Engineering Journal or Information Science. She has given keynotes or was invited speaker at the IV Workshop on Data & Artifact Centric BPM in Innsbruck, 5th International Workshop on Decision Mining & Modeling for Business Processes BPM in Barcelona, the X National Conference of BPM in Madrid, in the 28th IBIMA Conference, and in the biannual International Summer School on Fault Diagnosis of Complex Systems.

3.2.4 Empowering Conformance Checking using Big Data through horizontal decomposition

Published in the Information Systems journal (Vol. 99, p.101731). Elsevier BV.

- ***Authors**: Álvaro Valencia-Parra, Ángel Jesús, María Teresa Gómez-López, Josep Carmona, Robin Bergenthum.*
- ***DOI**: 10.1016/j.is.2021.101731.*
- ***Rating**: Q3 (JCR'20 2.309).*

Information Systems 99 (2021) 101731

Contents lists available at ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/is

Empowering conformance checking using Big Data through horizontal decomposition

Álvaro Valencia-Parra [a], Ángel Jesús Varela-Vaca [a,*], María Teresa Gómez-López [a], Josep Carmona [b], Robin Bergenthum [c]

[a] Departamento de Lenguajes y Sistemas Informáticos, Universidad de Sevilla, Spain[1]
[b] Department of Computer Science, Universitat Politècnica de Catalunya, Spain[2]
[c] Fakultät für Mathematik und Informatik, FernUniversität in Hagen, Germany[3]

ARTICLE INFO

Article history:
Received 1 March 2020
Received in revised form 17 November 2020
Accepted 24 January 2021
Available online 18 February 2021
Recommended by Gottfried Vossen

Keywords:
Conformance checking
Decompositional techniques
Big Data
MapReduce

ABSTRACT

Conformance checking unleashes the full power of process mining: techniques from this discipline enable the analysis of the quality of a process model through the discovery of event data, the identification of potential deviations, and the projection of real traces onto process models. In this way, the insights gained from the available event data can be transferred to a richer conceptual level, amenable for human interpretation. Unfortunately, most of the aforementioned functionalities are grounded in an extremely difficult fundamental problem: given an observed trace and a process model, find the model trace that most closely resembles to the trace observed. This paper presents an architecture that supports the creation and distribution of alignment subproblems based on an innovative horizontal acyclic model decomposition, disengaged from the conformance checking algorithm applied for their solution. This is supported by a Big Data infrastructure that facilitates the customised distribution of a gross amount of data. Experiments are provided that testify to the enormous potential of the architecture proposed, thereby opening the door to further research in several directions.

1. Introduction

By means of conceptual models, organisations tend to define complex business processes that must be followed to achieve their objectives [1]. Sometimes the corresponding processes are distributed across various systems, in which the majority of cases include human tasks, thereby inadvertently enabling the occurrence of unexpected deviations with respect to the (normative) process model. This is aggravated by the appearance of increasingly complex processes, where the observations are provided by heterogeneous sources, such as Internet-of-Things (IoT) devices involved in Cyber–physical Systems [2].

Conformance checking [3] techniques provide mechanisms to relate modelled and observed behaviour, so that the deviations between the footprints left by process executions and the process models that formalise the expected behaviour can be revealed.

One of the major challenges in conformance checking is the *alignment problem*: given an observed trace σ, compute an end-to-end model run that more closely resembles σ. Computing alignments is an extremely difficult problem, with a complexity exponential in the size of the model or the trace [4]. Intuitively, computing an alignment requires a search through the state space of the model which, in certain cases implies an extensive exploration when the process model is large and/or highly concurrent.

In order to face the challenge of computing alignments, the conformance checking community has proposed widely differing alternatives. Among these, we highlight decompositional techniques, which break the alignment problem into segments, whose solutions can be composed to reconstruct the final alignment [5–8]. All these decompositional approaches feature a common strategy which involves decomposing the problem by means of *vertical cuts* of the process model, and then projecting the traces in the log accordingly in order to derive subtraces that only contain events of the alphabet corresponding to each model fragment. Although, in very particular cases (e.g., well-structured process models), the aforementioned decompositional approaches represent a significant alleviation of the alignment problem, they

* Corresponding author.
E-mail addresses: avalencia@us.es (Á. Valencia-Parra), ajvarela@us.es (Á.J. Varela-Vaca), maytegomez@us.es (M.T. Gómez-López), jcarmona@cs.upc.edu (J. Carmona), robin.bergenthum@fernuni-hagen.de (R. Bergenthum).

[1] http://www.idea.us.es.
[2] https://www.cs.upc.edu.
[3] https://www.fernuni-hagen.de.

https://doi.org/10.1016/j.is.2021.101731
0306-4379/

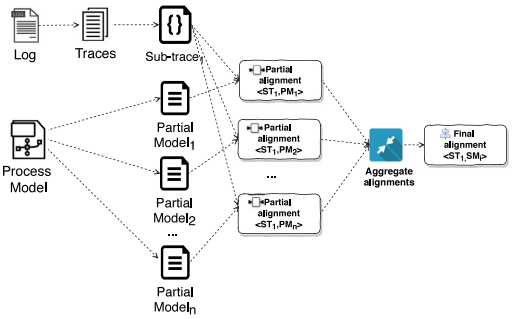


Fig. 1. Functional description of the Big Data architecture to compute alignments.

rely on very stringent conditions (e.g., model fragments should agree on the alphabet at the frontiers), and provide weak guarantees (e.g., necessary conditions for deriving an alignment), which hamper them from being applied in general.

In this paper, we step back from the decompositional approach, and focus on working at a more abstract, architectural level. As earlier mentioned, normally the complexity of computing the alignment problem has been addressed by means of the horizontal and vertical decomposition techniques [9]. We propose a Big Data infrastructure focused on a specific horizontal decomposition, which involves the unfolding of process models, and employing the MapReduce paradigm [10] for the decomposition and aggregation. At first sight, our functional strategy (depicted in Fig. 1) will not bring any new ideas to the landscape of decompositional techniques: the process model is decomposed into a set of partial models, and traces in the log are projected into subtraces. These two types of elements are then distributed and their partial solutions are composed to aggregate a final alignment. This distribution of the problem may facilitate the simplification of the problem, by splitting the conformance analysis into partial models (with smaller search spaces) and subtraces across several nodes (Map), and combining the partial alignments[4] obtained from different algorithms in the nodes (Reduce). The general idea of applying MapReduce for conformance checking is not new, as is analysed in the related work section. However, the Big Data [11] framework proposed in this paper is innovative for the following reasons:

- A new decomposition is proposed, which differs from the aforementioned approaches in one important feature: instead of a vertical cut, it is based on horizontal, end-to-end cuts that can be obtained by what we call *acyclic cover*, which originates from a partial order representation of the initial process model. The horizontal decomposition of the model limits the search space dividing the model into its possible execution models, and analysing the alignment with every trace. However, in the vertical decomposition, both traces and the model are fragmented, thereby reducing the search space for each part of the model and each trace fragment.
- The framework enables the construction, distribution, and parallelisation of the computing alignment between different nodes in a Big Data environment to be tuned in accordance with the features of the problem and the available requirements. Moreover, the application of heuristics is proposed to optimise the resolution of the subproblems.
- It enables us to choose and customise the conformance checking algorithm, by making it possible to compute the alignment with different techniques. In this case, we have used the A* algorithm as a classic solution, and the *Constraint Programming Paradigm* [12] as a new solution, in order to show how different types of alignment algorithms can be applied in the distributed paradigm.
- The development of a practicable infrastructure based on Big Data represents a leap forward in the resolution of conformance checking problems of a more complex nature, and reduces the resource limitations of the current solutions evaluated locally.

The paper is organised as follows: Section 2 analyses the related work. Section 3 includes the necessary foundations to understand the state of the art and the proposal. Section 4 determines how the use of Big Data techniques provides mechanisms for the partitioning and distribution of the computation of the conformance checking analysis. Section 5 describes how the A* algorithm and Constraint Programming can be applied to traces that represent the acyclic horizontal partial models. Section 6 depicts the experiments carried out to evaluate our proposal, and finally, Section 7 presents the conclusions.

2. Related work

The seminal work in [4] proposed the notion of alignment and developed a technique based on A* to compute optimal alignments for a particular class of process models. Improvements of this approach have been presented recently in various papers [13,14]. These approaches represent the state-of-the-art technique for computing alignments, and can be adapted (at the expense of a significant increase in the memory footprint) to provide all optimal alignments. Alternatives to A* have appeared in recent years: in the approach presented in [15], the alignment problem is mapped as an instance of *automated planning*. Automata-based techniques have also appeared [16,17]. The techniques in [16] (and recently extended in [18]) rely on state-space exploration and determination of the automata corresponding to both the event log and the process model, whilst the technique in [17] is based on computing several subsets of activities and projecting the alignment instances accordingly. In spite of the significant progress made, the aforementioned techniques still have problems in dealing with large inputs.

The work in [19] presents the notion of *approximate* alignment to alleviate computational demands by proposing a recursive paradigm on the basis of the structural theory of Petri nets. In spite of its resource efficiency, the solution is not guaranteed to be executable. Alternatively, the technique in [20] presents a framework to reduce a process model and the event log accordingly, with the goal of alleviating the computation of alignments. The obtained alignment, called *macro-alignment* since some of the positions are high-level elements, is expanded based on the information gathered during the initial reduction. Techniques using a local search have also recently been proposed [21]. Although the approximate techniques can provide solutions where exact/optimal techniques fail, they only provide certain guarantees for very restricted classes of models.

Against this background, the process mining community has focused on dividing and conquering the problem of computing alignments as a valid alternative to this problem, with the aim of alleviating its complexity without degrading the quality of the solutions found. Our focus now turns to decompositional

[4] *Partial alignment* means the computation of the alignment for a partial model and a subtrace.

approaches towards the computation of alignments, which are more closely related to the research of this paper.

Decompositional techniques have been presented [5–7] which, instead of computing optimal alignments, focus on the *crucial problem* of whether a given trace fits a process model. These techniques vertically decompose the process model into pieces that satisfy certain conditions. Therefore, only *valid* decompositions [5], which satisfy restrictive conditions on the labels and connections forming a decomposition, guarantee the derivation of a real alignment. The notion of *recomposition* has since been proposed on top of decompositional techniques, in order to obtain optimal alignments whenever possible by modifying the decomposition (typically by merging sets) when the required conditions are not met [8]. In contrast to the aforementioned vertical decomposition techniques, our methodology does not require this last modification of partial solutions, and therefore can provide a fast alternative to these methods at the expense of losing the guarantee of optimality.

There has also been related work on the use of partial order representations of process models for the computation of alignments. In [22], unfoldings are employed to capture all possible transition relations of a model so that they can be used for online conformance checking. In contrast, unfoldings were used recently in a series of papers [23,24] to significantly accelerate the computation of alignments. We believe that these approaches, especially the latter two, can easily be integrated into our framework.

The work of [18] can also be considered a decompositional approach since it proposes decomposing the model into sequential elements (*S-components*) so that the state-space explosion of having concurrent activities is significantly alleviated. This work is compatible with the framework suggested in this paper since the model restrictions assumed in [18] are satisfied by the partial models arising from our horizontal decomposition.

Finally, the MapReduce distributed programming model has previously been considered for process mining. For instance, Evermann applies it to process discovery [25], whilst [26] applies it for monitoring declarative business processes.

3. Foundations

We denote $\perp$ as the empty set. Let A be a set of elements, and we denote A^* as the set of all sequences over elements of A. Let $a, b \in (A \cup \{\perp\})^*$ be two sequences. We denote $a^{\perp}$ as the sequence a, but omit all elements $\perp$ from a. We write $a \mathrel{\widehat{=}} b$ if $a^{\perp} = b^{\perp}$ holds.

3.1. Process models

In this paper, we describe process models and partial models by means of labelled Petri nets.

Definition 1 (*Labelled Petri Net*)**.** A labelled Petri net is a tuple (P, T, F, Σ, ℓ) where P and T are finite disjoint sets of places and transitions, respectively, $F : (P \times T) \cup (T \times P) \to \{0, 1\}$ is the flow-relation, Σ is the alphabet, and $\ell : T \to \Sigma \cup \{\perp\}$ is the labelling function.

Fig. 2 depicts a labelled Petri net. Places are represented by circles and transitions by rectangles. Every transition has a unique name and a label on top. Places and transitions are connected in accordance with the flow-relation.

In Petri nets, there is the so-called firing rule. Transitions of a Petri net can be fired, thereby changing the state of the net.

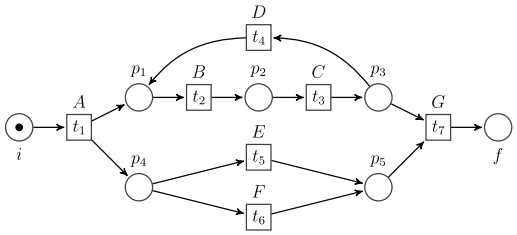


Fig. 2. A labelled Petri net.

Definition 2 (*Firing Rule*)**.** Let $N = (P, T, F, \Sigma, \ell)$ be a labelled Petri net. A function $m : P \to \mathbb{N}_0$ is a marking of N. We define, $\bullet t : P \to \{0, 1\}$ as $\bullet t(p) := F(p, t)$, and $t\bullet : P \to \{0, 1\}$ as $t \bullet (p) := F(t, p)$. A transition $t \in T$ is enabled at marking m if $m \geq \bullet t$ holds. If transition t is enabled, then transition t can be fired. In this case, we write $m\,[t\rangle$. Firing t changes the marking m to $m' := m - \bullet t + t\bullet$. In this case, we write $m\,[t\rangle\, m'$.

We depict a marking by putting black dots, called tokens, in the places of the marking. For example, Fig. 2 depicts the initial state of the labelled Petri net. The initial marking only contains place i once. In this marking, only transition t_1, labelled as A, is enabled. Firing t_1 leads to the marking where i does not carry a token, and both places in the post-set of t_1 each carry a token.

Starting at the initial marking, sequentially enabled sequences of transitions are words of the language of the Petri net. The related traces of labels are the so-called trace-language.

Definition 3 (*Language of a Petri Net*)**.** Let $N = (P, T, F, \Sigma, \ell)$ be a labelled Petri net. A marked Petri net is a tuple (N, m_0, m_f) where m_0 is the initial marking and m_f is the final marking. A sequence $\langle t_1, \ldots, t_n\rangle \in T^*$ is a firing sequence. If there is a sequence of markings $\langle m_1, \ldots, m_{n+1}\rangle$ such that $m_1\,[t_1\rangle\, m_2$, $m_2\,[t_2\rangle\, m_3$, …, $m_n\,[t_n\rangle\, m_{n+1}$ holds, we can write $m_1\,[t_1, \ldots, t_n\rangle\, m_{n+1}$.

$$\mathcal{L}(N) := \{\langle t_1, \ldots, t_n\rangle \in T^* \mid m_0\,[t_1, \ldots, t_n\rangle\, m_f\}$$
$$\mathcal{T}(N) := \{\sigma \in \Sigma^* \mid \langle t_1, \ldots, t_n\rangle \in \mathcal{L}(N) \wedge \sigma \mathrel{\widehat{=}} \langle \ell(t_1), \ldots, \ell(t_n)\rangle\}$$

$\mathcal{L}(N)$ is the language of N; $\mathcal{T}(N)$ is the trace-language of N.

In Fig. 2, if we assume the final marking where only place f carries one token, for example $\langle t_1, t_2, t_3, t_6, t_7\rangle$ and $\langle t_1, t_2, t_3, t_5, t_4, t_2, t_3, t_7\rangle$ are words of the language, and $\langle A, B, C, F, G\rangle$ and $\langle A, B, C, E, D, B, C, G\rangle$ are the related traces.

3.2. Conformance checking

Event logs record the behaviour observed from the execution of a business process.

Definition 4 (*Trace, Event Log*)**.** Let Σ be an alphabet. A sequence $\sigma \in \Sigma^*$ is a trace. A multi-set of traces $L : \Sigma^* \to \mathbb{N}_0$ is an event log.

The classic notion of aligning an event log and a process model was introduced by [4]. An alignment maps a trace of an event log to a firing sequence of the model. An alignment replays the trace and the firing sequence simultaneously, where either the trace, the firing sequence, or both move. Trace and sequence are allowed to move synchronously only if the label of the transition matches the event.

We consider the Petri net depicted in Fig. 2 with initial state i and final state f. By aligning the Petri net to the trace $\langle A, A, B, C,$

$D, B, C, G\rangle$ for instance, we obtain numerous possible alignments, where moves of the trace are at the top, and moves of the model are at the bottom of a table.

A	A	B	C	D	B	C	$\perp$	G
t_1	$\perp$	t_2	t_3	t_4	t_2	t_3	t_5	t_7
A	A	B	C	D	B	$\perp$	C	G
t_1	$\perp$	t_2	t_3	t_4	t_2	t_5	t_3	t_7
A	A	B	C	D	B	$\perp$	C	G
$\perp$	t_1	t_2	t_3	t_4	t_2	t_6	t_3	t_7

...

Definition 5 (*Alignment*)**.** Let $N = (P, T, F, \Sigma, \ell, m_0, m_f)$ be a marked Petri net, σ be a trace of an event log $L : \Sigma^* \to \mathbb{N}_0$, and $\tau \in \mathcal{L}(N)$ be a firing sequence. The set

$$\mathcal{M} \coloneqq \{(a, t) \in (\Sigma \times T) | \ell(t) = a\} \cup (\Sigma \times \{\perp\}) \cup (\{\perp\} \times T)$$

is the set of legal moves. An element $\langle (a_1, t_1), \ldots, (a_n, t_n)\rangle \in \mathcal{M}^*$ is an alignment of σ and τ iff $\langle a_1, \ldots, a_n\rangle \mathrel{\widehat{=}} \sigma$ and $\langle t_1, \ldots, t_n\rangle \mathrel{\widehat{=}} \tau$ holds.

We define a cost-function to attain a cost for every alignment. Every move of an alignment adds to its cost, where asynchronous moves add greater cost than synchronous moves [4].

Definition 6 (*Cost-Function*)**.** Let $N = (P, T, F, \Sigma, \ell)$ be a labelled Petri net and let $L : \Sigma^* \to \mathbb{N}_0$ be an event log. Let $0 \leq \delta_1 < \delta_2, \delta_3$ hold. We define the cost-function $\lambda_{\delta_1,\delta_2,\delta_3} : \mathcal{M}^* \to \mathbb{N}_0$ as follows: for every alignment $\alpha = \langle (a_1, t_1), \ldots, (a_n, t_n)\rangle \in \mathcal{M}^*$, we define

$$\begin{aligned}\lambda(\alpha)_{\delta_1,\delta_2,\delta_3} \coloneqq\ & \delta_1 \cdot |\{(a, t) \in \alpha \mid a = \ell(t)\}| \\ & + \delta_2 \cdot |\{(a, \perp) \in \alpha\}| \\ & + \delta_3 \cdot |\{(\perp, t) \in \alpha\}|\end{aligned}$$

We fix a cost-function to calculate a so-called optimal alignment between a trace of an event log and a process model. An optimal alignment is an alignment with a lowest cost. In the previous example, if we define the cost of an asynchronous move as 1 and the cost of a synchronous move as 0, then the depicted alignments have a cost of 2.

4. Computing conformance checking with big data

4.1. Overview of the approach

The fundamental problem in conformance checking involves aligning a trace concerning a process model [3]. This problem, known as *the alignment problem*, is a search (which can be highly time-consuming) to find a model trace similar (according to a cost-function) to the observed trace. Please refer to Section 2 for a complete overview of the current approaches for computing alignments.

Derived from the complexity of the alignment problem, we present a solution based on the creation of simpler problems that can be distributed on a Big Data architecture that aims to facilitate the computation of alignments on a grand scale. In this paper, we assume both process models and logs can be decomposed so that we can take advantage of a Big Data infrastructure, and therefore the fundamental problem of computing an alignment can be distributed over the infrastructure in a MapReduce fashion [10]. As will be observed see in Section 5, to instantiate the architecture for a real situation, we build upon our previous work [12] and the case of a partial order decomposition of a process model (see Section 4.2). However, while the architecture presented in this section is not tied to any particular conformance checking algorithm, the decomposition technique must be based on the extraction of subtraces and the unfolding of a process model into partial models through horizontal decomposition. Other decompositional approaches available in the literature [5–7] might be employed, but lead to changes in the way in which the *Generate partitions of Problems*, *Map*, and *Reduce* activities are implemented. Furthermore, it should be borne in mind that other decompositional approaches must be capable of forming partitions so that these can be distributed among the nodes of the cluster.

In order to determine each of these parameters that describe how the subproblems are created, distributed, solved, and combined, Fig. 3 summarises the workflow followed in our approach. Since our proposal is not hooked to a specific alignment algorithm, it has been tested with two very different algorithms to analyse how the type of conformance technique algorithms can affect the Map and Reduce stages. In the first phase, the alignment algorithm is determined, as are the subtraces[5] and partial models. These are obtained through the unfolding process, which applies a horizontal decomposition technique. Once these aspects are defined, a subtrace and partial model pre-processing are needed (see *Pre-process traces and partial models*) to determine certain features used in the heuristics for the subsequent problem distribution. The system is then set up (see *Setting parameters and heuristics*) in terms of the number of partitions (set of alignment subproblems) to be distributed in each node, the subproblem assignations to each node according to the parameters obtained from the previous activity, the thresholds of time used for solving each subproblem, and the threshold of memory in the nodes of each cluster. When the parameters are configured, the MapReduce paradigm can be applied following the following three activities: *Generate partitions of problems*, *Map — Distribution and compute partitions*, and *Reduce — Combine results*. These are given in detail in Sections 4.2, 4.3, and 4.4, respectively. The framework follows the idea of the MapReduce paradigm as depicted in Fig. 4. The input of the problem is the set of alignment problems formed of a combination of a subtrace and a partial model. These alignment problems will be distributed in different divisions solved in each node, where the *Map* function is applied obtaining a map $\langle key, value\rangle$ whose key is the trace and the value is the alignment found for the set of traces involved in this subproblem. All the partial solutions represented by maps are then combined.

Our framework provides a mechanism to set up the parameters to perform the alignment analysis in a more efficient way. Therefore, after a solution is found, the parameters (i.e., timeout and number of partitions) can be adjusted to re-execute the alignment analysis, thereby reducing the time.

4.2. Generate partitions of problems

As indicated in Section 1, we aim to alleviate the complexity of a conformance checking problem by dividing a model into a set of partial models. A partial model covers a part of the behaviour of the original model. Furthermore, a partial model needs to be acyclic and conflict-free. Finally, it should be borne in mind that the approach assumes that partial models are generated through horizontal decomposition.

[5] As the reader will identify later, in this paper we use the term *subtrace* to stress the fact that the methodology proposed is general, although, in our particular explanations, subtraces will be full traces.

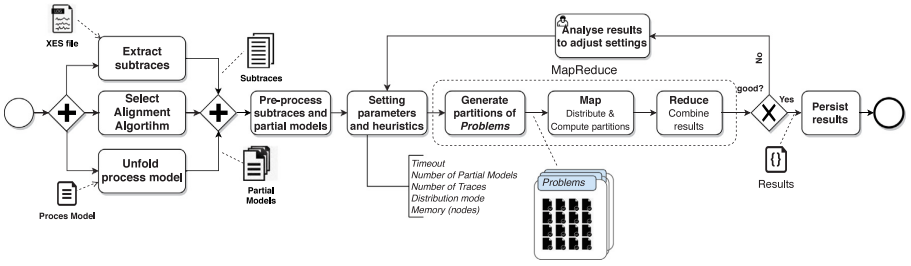


Fig. 3. Workflow of the approach.

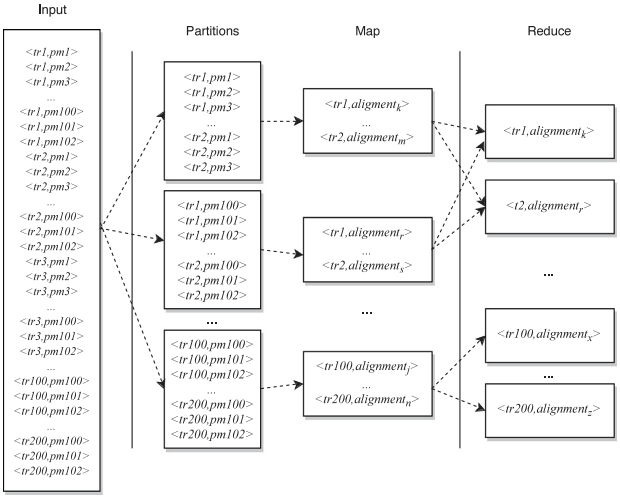


Fig. 4. MapReduce for alignment analysis.

Definition 7 (*Partial Model, Cover*)**.** Let $N = (P, T, F, \Sigma, \ell, m_0, m_f)$ and $N' = (P', T', F', \Sigma', \ell', m'_0, m'_f)$ be two marked Petri nets. N' is conflict-free iff $(m\,[t_1\rangle\, m' \wedge m\,[t_2\rangle) \implies m'\,[t_2\rangle$ holds. We call N' a partial model of N iff N' is conflict-free, acyclic, and $\mathcal{T}(N') \subseteq \mathcal{T}(N)$ holds. We call a set of partial models $\{N_1, \ldots, N_n\}$ a cover of N iff $\bigcup_i \mathcal{T}(N_i) = \mathcal{T}(N)$ holds.

In this respect, a partial alignment is the computation of the alignment of a partial model.

Fig. 5 depicts a partial model of Fig. 2. The depicted marked Petri net is conflict-free, acyclic, and its trace-language is $\{\langle A, B, C, E, G\rangle, \langle A, B, E, C, G\rangle, \langle A, E, B, C, G\rangle\}$. Obviously, this trace-language is a sub-set of the trace-language of Fig. 2. Fig. 6 depicts another partial model. In this example, transitions t_2 and t_5 carry the label B, while transitions t_3 and t_6 carry the label C. Thus, the loop of Fig. 2 is unfolded.

One straightforward approach to splitting a Petri net into a cover is to calculate its branching process [27]. It is well-known that the set of so-called process nets of a branching process is a cover. It should be borne in mind that the branching process itself can be infinite, but given the maximal length of a trace of the log, we can always determine a sufficient depth to calculate an appropriate prefix of a cover for the alignment problem at hand. In the literature, there is a rich body of approaches to calculating finite representations of an infinite branching process in a reasonable time [28].

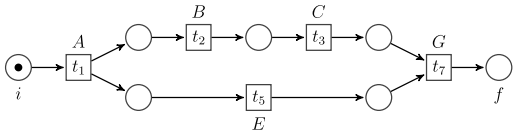


Fig. 5. A partial model of Fig. 2.

Fig. 7 depicts a prefix of the branching process of the model of Fig. 2. This acyclic labelled net is able to execute all traces up to length seven of the original model. It is a prefix because the looping behaviour of transitions B, C, and D generate infinite behaviour. In a branching process, a model is unfolded so that all places have at most one preceding transition, for instance, the two places (see p_5 and p_{10}) behind transitions labelled by E and F. In Fig. 2, this pair is only one place (see p_5). The same holds for all places before transitions labelled by B and places behind transitions labelled by D. In the original model, they are just one place (see p_1 in Fig. 2). Thus, conflicts and cycles are unfolded. Every connected subnet of a branching process, whereby all places have at most one subsequent transition, is called an occurrence net. For example, the set of transitions $\{t_1, t_2, t_3, t_5, t_7\}$, with all connected places, form the partial model of Fig. 5. Transitions $\{t_1, t_2, t_3, t_4, t_6, t_8, t_{10}, t_{13}\}$ are the partial model of Fig. 6.

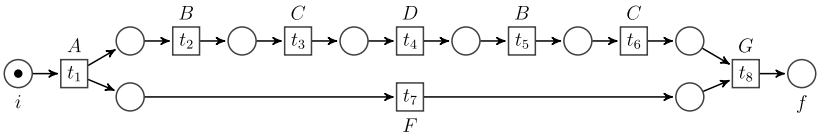


Fig. 6. Another partial model of Fig. 2.

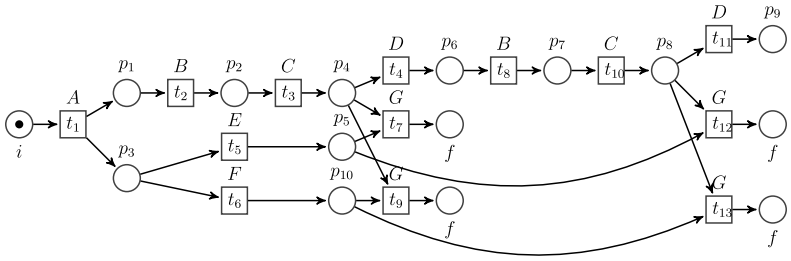


Fig. 7. A prefix of the branching process of Fig. 2.

The use of occurrence nets of branching processes constitutes only one of many possibilities for the horizontal decomposition of a model. Log-based unfolding [29] and token flow-based unfolding [28] can generate similar decompositions. In the more general setting of the paper, the set of partial models is required to be given as a set of acyclic, conflict-free marked Petri nets.

Every trace of a partial model can be replayed by its original model. Thus, for every alignment of the partial model, there is a related alignment in the original model that incurs the same cost. For example, the firing sequence $\langle t_1, t_2, t_3, t_4, t_5, t_7, t_6, t_8\rangle$ of the partial model depicted in Fig. 6 can be replayed by the original model depicted in Fig. 2 by $\langle t_1, t_2, t_3, t_4, t_2, t_6, t_3, t_7\rangle$. Obviously, related (replayed) alignments have the same cost:

A	A	B	C	D	B	$\perp$	C	G
$\perp$	t_1	t_2	t_3	t_4	t_5	t_7	t_6	t_8
$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$
$\perp$	t_1	t_2	t_3	t_4	t_2	t_6	t_3	t_7

If a set of partial models covers a Petri net, then, for every alignment of the original model, there exists a partial model covering each alignment. This holds true since the set of partial models can replay every trace of the original model. An optimal alignment can be therefore be calculated for the original model by calculating an optimal alignment for the set of partial models.

The division of the problem into smaller partitions forms the base of the application of the MapReduce paradigm. It is therefore necessary to tackle the problem of partitioning an alignment problem (AP) into a set of subproblems by distributing the set of traces of an event log and the set of partial models extracted from a process model. Firstly, and Fig. 3, the process model and log are decomposed into subtraces and partial models that can be analysed independently, thereby obtaining the alignment in a more efficient way.

Definition 8 (*Decomposition, Alignment Subproblem*)**.** Let AP be an alignment problem aligning a set of traces $Tr = \{tr_1, tr_2, \ldots, tr_n\}$ to a model M. Let $Pm = \{pm_1, pm_2, \ldots, pm_m\}$ be a cover of M. We call every element $pr \in (Tr \times Pm)$ an *alignment subproblem*. We write $AP = (Tr \times Pm)$ and call $(Tr \times Pm)$ a *decomposition* of AP into $n \cdot m$ subproblems.

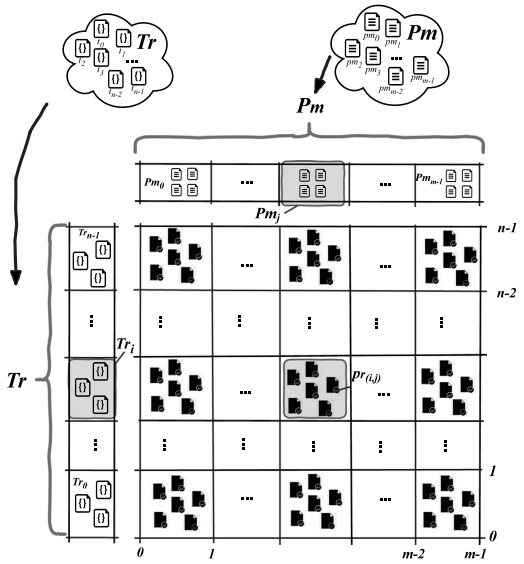


Fig. 8. Partitions and sets of subproblems.

In an ideal scenario with unlimited resources, each alignment subproblem could be solved independently and in parallel. In this case, the total run-time needed to solve AP would be the time spent on the most complex subproblem plus the time spent combining the partial alignments. Here, we would need as many nodes as subproblems to process all subproblems in parallel.

In real-life applications, the number of subproblems is much too high to simply generate a node for every problem. Thus, subproblems need to share nodes. To control the distribution of subproblems to nodes, the set of all possible subproblems is partitioned into groups of subproblems that share the same features. Features are made up of the involved trace and partial alignments

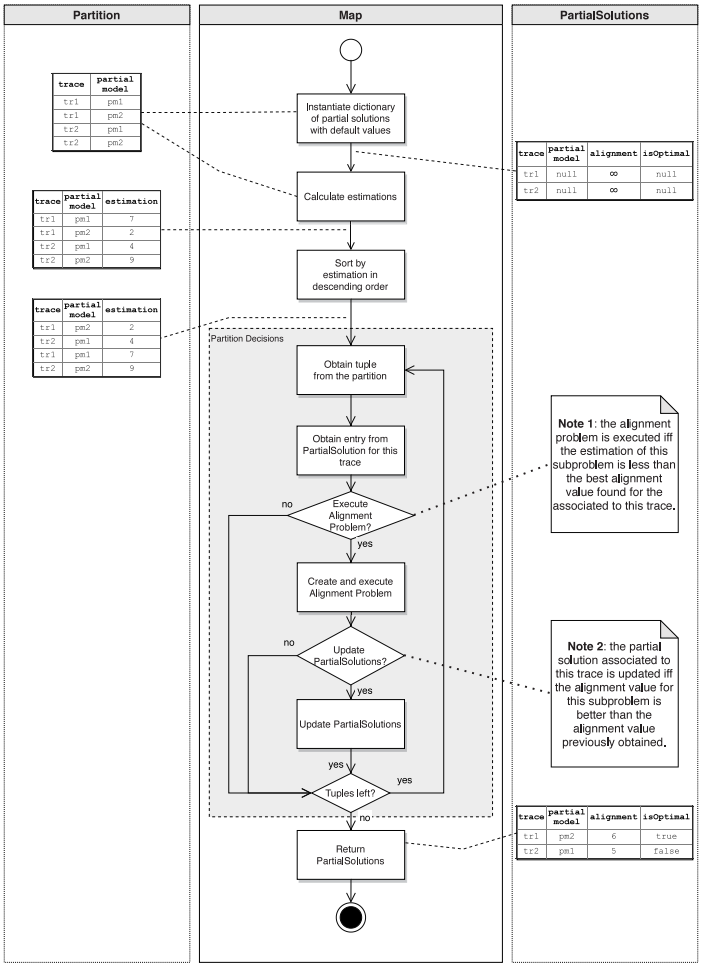


Fig. 9. Activity diagram to describe the Map algorithm.

calculated in other subproblems. How to properly group and distribute subproblems to calculate solutions efficiently is analysed below.

Definition 9 (*Partitions*)**.** Let Tr be a set of traces. We call a set of disjoint sets of traces $\{Tr_1, Tr_2, \ldots, Tr_n\}$ a *partition* of Tr if $Tr = \bigcup_{i=1}^{n} Tr_i$ holds. Let Pm be a set of partial models. We call a set of disjoint sets of partial models $\{Pm_1, Pm_2, \ldots, Pm_m\}$ a *partition* of Pm if $Pm = \bigcup_{i=1}^{m} Pm_i$ holds. Let Tr_i be a set of the partition of Tr and let Pm_j be a set of the partition of Pm. We call $pr_{(i,j)} \coloneqq (Tr_i \times Pm_j)$ a partition of the alignment subproblems.

Partitions $pr_{(i,j)}$ define sets of alignment subproblems. When each alignment subproblem $(Tr_i \times Pm_j)$ is solved, a partial alignment is obtained. Figure schematically 8 depicts partitions of Tr and Pm and the resulting sets of alignment subproblems.

In the next subsection, we will discuss the distribution of every partition of alignment problems following the MapReduce strategy [10], which is a programming model to support parallel computing for large collections of data.

4.3. Map - Distribute and compute alignment problem partitions

The Map function is based on solving smaller problems, thereby obtaining partial solutions for their subsequent combination. The algorithm used in the map function is represented in Fig. 9. It receives a partition of subproblems and creates a dictionary of *partial solutions* with default values. For each subproblem, it then makes a lower-bound estimation for the possible alignment that can be taken before it is solved. This estimation is employed to sort the subproblems to solve in the same partition (sequentially solved). The estimation is obtained by comparing model and trace: (1) checking the size of the trace w.r.t. the maximum number of events that can be extracted from the submodel (e.g., if the trace has 100 events and the longest trace generated by the submodel is 90, then the alignment (see Definitions 5 and 6) must be at least 10); (2) the events that occur in the trace but not in the model and vice versa; and (3) considering the number of occurrences of events with regard to the submodel (e.g., if the event A is repeated three times in the

trace but only twice in the submodel, then the alignment cost must be at least 1). These values are calculated and aggregated to generate an estimation as the lowest value that the alignment can take. If an alignment subproblem discovers better alignment than the estimation for the same trace, then it is illogical to evaluate the remaining subproblems with an inferior estimation. The partition of subproblems is then sorted by estimation in ascending order. Sorting is crucial for the optimisation of the execution time since it prevents the alignment process from executing subproblems that would fail to provide an improvement of the best alignment found up to that moment (see Note 1 in Fig. 9). If a new alignment is obtained, then the partial solution associated with that trace is updated if the new alignment value is better than the previous value (see Note 2 in Fig. 9). Note that the partial solutions have an attribute called *isOptimal*. This will be true when it is possible to guarantee that the solution associated with this trace is the optimal solution (note that the notion of non-optimality is introduced due to the existence of a timeout, which prevents the search space from being searched in its entirety). If the isOptimal attribute of any of the subproblems that were executed is marked as *false* due to the timeout being reached, then we cannot guarantee that the solution to any other subproblem associated to that trace is the optimal solution, because any other subproblem with a better previously executed estimation value could have returned a better alignment value if the timeout had not been reached.

In order to illustrate the algorithm, Fig. 10 presents the iteration of the *partition* presented in Fig. 9. At this point, it should be borne in mind that the partitions are already sorted by estimation. There are four elements to process, and hence, there are four iterations. In *Iteration 1*, the subproblem $\langle tr1, pm2\rangle$ is processed. The alignment process is then executed because the estimation yields a value of 2, which could improve the partial solution found until the moment (∞). Once the alignment value (6) has been computed, the partial solution for $tr1$ $\langle tr1, pm2, 6, true\rangle$ is stored. In *Iteration 2*, we have a similar situation with $\langle tr2, pm1\rangle$, where the partial solution is also updated after obtaining an alignment of 5. However, the timeout was fired, and therefore the alignment cannot be guaranteed to be optimal ($\langle\langle tr2, pm1, 5, false\rangle\rangle$). However *Iteration 3* does not execute the alignment process nor update the partial solution previously obtained for $tr1$, since the estimation for the subproblem $\langle tr1, pm1\rangle$ is greater than the best optimal computed alignment.

In *Iteration 4*, the same situation arises, and hence the partial solution formerly found for $tr2$ is not updated.

4.4. Reduce - Combining alignment problem result

The Reduce phase is responsible for combining the partial solutions that are generated during the Map phase. Each partition yielded a set of partial solutions, with the following information: trace, partial model, alignment value, and an indicator pointing out whether the solution is optimal or if it is impossible to ensure that the alignment obtained is the optimal solution. In this phase, all the partial solutions corresponding to the same trace are combined, and that with the best alignment value is selected as the best solution for such a trace.

Fig. 11 depicts the Reduce process, and includes some partial solutions to show the execution. The proposal is based on the function known as *reduceByKey*, which groups data in terms of the key of the data provided by the map function, and then applies a function in order to combine the values associated with each key. For the alignment problem, the Reduce phase groups the partial solutions with the same key (i.e., with the same trace), and combines every partial solution in the same group, thereby obtaining another partial solution. Following the example of Fig. 11, the tuples of *Partial Solutions* 1 and 2 are grouped according to their trace ($tr1$ and $tr2$). The tuples in each of these groups are combined returning a single tuple in each case. The new obtained partial solution follows the form:

- **trace:** the trace that was employed to create the groups and is shared by every tuple in the group.
- **partial model:** the partial model whose alignment is minimal for that trace.
- **alignment:** the minimal alignment of every tuple.
- **isOptimal:** the $\wedge$ combination of the *isOptimal* values of every tuple. This means that if it is *false* for a tuple, then the other tuples related to the same trace will also be marked as *false*, since it is impossible to ensure that the found alignment is optimal because the problem has not been fully analysed.

5. Interchangeable solutions for encoding alignment

The MapReduce algorithm presented in the previous section can be applied to various types of alignment techniques, subtraces, and partial models. Several algorithms that have tackled the conformance checking problem in the context of business processes (see Section 2 for a full description). In this section, two such algorithms are included: the A* algorithm as an example of a classic algorithm developed by other authors [4]; and a new implemented solution based on the Constraint Programming Paradigm. These two algorithms have the same objective (i.e., to discover the alignment). However, we have included the Constraint Programming Paradigm since it enables certain special features to be incorporated, such as restricting the domain of the possible value where the alignment can be found, and determining a maximum time of resolution per subproblem that returns the best solution found up until the timeout.

5.1. Alignment based on the A* algorithm

One of the most relevant solutions to computing alignments found in the literature is the A* algorithm [4]. It has been successfully employed as a feasible approximation to discover the optimal alignment between the process model and traces [3]. Essentially, the model and trace are combined into a synchronous product. Fig. 12 illustrates the synchronous product, and presents the partial model, obtained from a cover (see Definition 7) given in Fig. 6, and the log *trace*: $\langle A, B, E, D, C, B, C, F, G\rangle$.

The simplest way to compute alignment is to build the reachability graph (see Definition 7, [3]) from the synchronous product, and then to deduce the shortest path from the initial marking to the final marking. However, the construction of the full reachability graph is not always possible due to the state space explosion problem. To overcome this problem, the reachability graph is built in pieces. The A* algorithm is efficiently used (see Chapter 7.3, Procedure 2 [3]) to compute the shortest path. The core of the A* algorithm relies on a heuristic function, $f(m) = g(m) + h(m)$, which guides the search, where $g(m)$ is the cost of the path from the initial marking to m. For instance, for any reachable state m, A* must determine $h(m) \leq h^*(m)$, where $h^*(m)$ is the shortest path from m to the final marking. There are cases in which A* fails to compute the alignments since it is highly complex and time-consuming (e.g., in models with a very high levels of parallelism [18]).

Our approach integrates the implementation of the A* algorithm provided by the Python library *PM4Py*.[6]

[6] PM4Py: https://pm4py.fit.fraunhofer.de/.

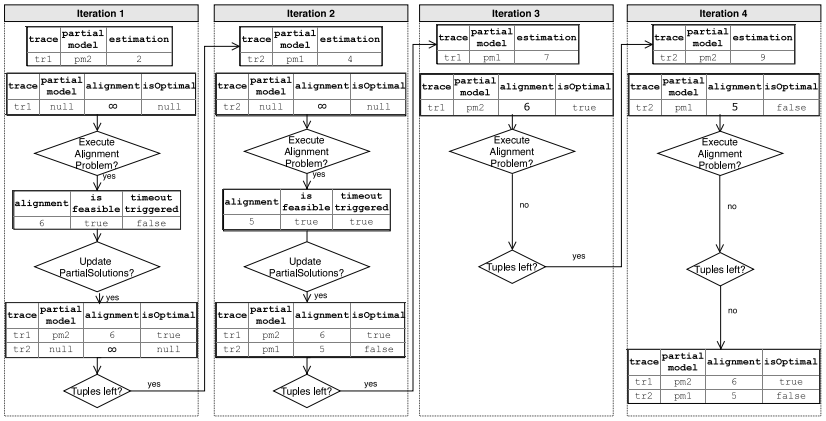


Fig. 10. Execution trace of the Map function.

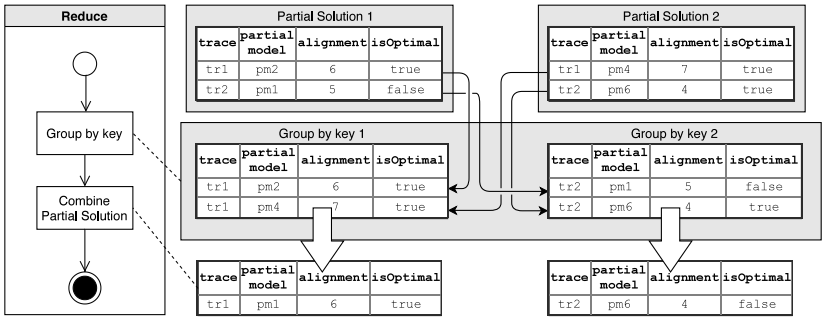


Fig. 11. Execution trace of the Reduce function.

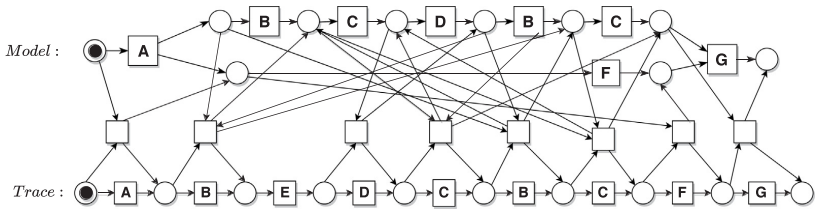


Fig. 12. Example of synchronous product of model and trace.

5.2. Alignment based on constraint programming

The Constraint Programming paradigm is a general-purpose technique that can be applied to optimise problems. Since the alignment problem is an optimisation problem that can be distributed, the incorporation of this new solution in our framework is considered to be relevant as an evolution of a previous proposal [12]. Moreover, since the alignment computation can be modelled as a variable and restrictions whose domain can be bounded, and thanks to the decomposition of the model into submodels, we consider it relevant to analyse how the former resolution of subproblems can be used to tighten the possible domain for the analysis in further resolutions. The partial model, obtained from a cover (see Definition 7) given in Fig. 6, and the log *trace*: $\langle A, B, E, D, C, B, C, F, G\rangle$, are used as a running example to illustrate the encoding based on Constraint Programming. The partial model can contain concurrent paths, that is, there would be *and*-splits that divide the execution into various branches that can be executed in parallel.

In our approach, the computation of the alignment problem of a log trace and a partial model is encoded as a Constraint Problem for the reduction in running time and resources of the proposal presented in [12]. Thus, the information extracted from the partial model and the trace, such as the name of transitions, events, the execution, order and their possible positions, are translated into variables, constraints, and an objective function of a Constraint Optimisation Problem (COP). The horizontal decomposition can ensure that the resulting partial models contain

or-splits. In fact, this helps reduce the complexity of the COP regarding the number of restrictions and the number of variables and the domain of the variables.

5.2.1. Constraint and optimisation problems in a nutshell

Constraint Programming is a paradigm that permits the declarative description of the constraints that determine a problem [30, 31]. Constraint Programming brings together a set of algorithms to determine the solutions of a problem described.

Definition 10 (*The Constraint Satisfaction Problem*)**.** (CSP) is defined by a 3-tuple $\langle X, D, C\rangle$, where $X = \{x_1, \ldots, x_n\}$ is a finite set of variables, $D = \{d(x_1), \ldots, d(x_n)\}$ is a set of domains of the values of the variables, and $C = \{C_1, \ldots, C_m\}$ is a set of constraints. Each constraint C_i determines relations R between a subset of the variables $V = \{x_i, x_j, \ldots, x_l\}$.

A constraint $C_i = (V_i, R_i)$ simultaneously specifies the possible values of the variables in V that satisfy R. Let $V_k = \{x_{k_1}, \ldots, x_{k_l}\}$ be a subset of X, and an l-tuple $(x_{k_1}, \ldots, x_{k_l})$ from $d(x_{k_1}), \ldots, d(x_{k_l})$ can therefore be called an *instantiation* of the variables in V_k. An instantiation is a solution iff it satisfies the constraints C. The CSP solvers enable one tuple of instantiation of one, multiple, or all these values to be sought in accordance with the requirement of the problem.

An example of its applicability in the alignment context is given by its representation of the order relation existing in the models and the traces, as found in Fig. 6. By using a set of variables to represent the order of the events, and by satisfying the relative constraints of the activities that appear in the partial model, the alignment can be encoded in the following CSP.

// Variables
$position_A$, $position_B$, $position_C$ ... in the domain {0..trace.length-1}
// Constraints of the log trace
$position_A == 0$
$position_C == 1$
$position_B == 2$
$position_{AD} == 3$
...
$position_{AZ} == 14$
// Constraints of the partial model
$position_A < position_C$
$position_C < position_{AC}$
$position_C < position_{AF}$
$position_{AC} < position_{AD}$
...

If the model and the event cannot be aligned, this CSP will not be satisfied. However, no more feedback regarding the level of misalignment is provided by the resolution of the CSP. In this case, a Constraint Optimisation Problem (COP) is able to ascertain the minimal distance between the partial model and the log observed since a COP is a CSP that includes an optimisation function. Only the solution of the CSP that satisfies the optimal function can be the solution of the COP.

Constraint Optimisation Problems (COPs) have already been employed to detect the alignment between the expected and the observed behaviour in model-based diagnosis [32,33], specifically when the behaviour is described by means of business process models [34–36]. These studies used the concept of *reified constraints* as a mechanism to assign a Boolean value to the constraints included in the model [34], whereby a constraint that cannot be satisfied during the CSP resolution can be relaxed. Since the idea is to determine the minimal distance between the model and the log, these relaxed constraints must be the minimum number, defined as the objective of the function to be optimised.

Following the previous example, the COP below is created where the *Ref* variables relate to the *reified constraints*.

// Variables
Ref_A, Ref_C, Ref_C ... in the domain {0..1}
$position_A$, $position_B$, $position_C$ ... in the domain {0..trace.lenght-1}
// Constraints of trace
$position_A == 0$
$position_C == 1$
$position_B == 2$
$position_{AD} == 3$
...
$position_A == 14$
// Constraints of the model
$Ref_A \wedge Ref_C \implies (position_A < position_C)$
$Ref_C \wedge Ref_{AC} \implies (position_C < position_{AC})$
$Ref_C \wedge Ref_{AF} \implies (position_C < position_{AF})$
$Ref_{AC} \wedge Ref_{AD} \implies (position_{AC} < position_{AD})$
...
maximize$(Ref_A + Ref_C + \ldots + Ref_{AF})$

Although the idea of the COP modelling follows the previous COP, in the following subsection, we approach the definition included in Section 3 in relation with a COP to determine the alignment between a partial model and a log trace.

5.2.2. Constraint optimisation problem for solving an alignment subproblem

Our proposal builds the COP from the perspective of the placement of the events in a positional order that satisfies both the log trace order and that of the partial model. However if this is not possible, then a number of the constraints are ignored from the COP firing reified constraints. The structure of the COP is as shown in Fig. 13.

As defined above, a COP is composed of a set of variables, a set of constraints, and an objective function. It is important to take into account the possibility that an event can appear more than once in a log trace derived, for example, from an unfolding process. In this case, a relabelling of the events is necessary to differentiate the variables that represent one or another , although some constraints must be included to express that they can represent the same transition. In detail, a COP is formed of:

- *Variables for the Log Events*: for each event in the log trace, two variables are created:
  - *Position (pos):* Integer variable with a domain between 0 and the number of events, that is, all the different locations of the events. This domain represents all the possible positions with respect to the partial model. In the running example, all the variables receive a domain from 0 to 8 since 9 is the total number of events, although the event E is not in the partial model and the transition H in the partial model is not included in the events.
  - *Deviation (dev):* Boolean variable which represents the correct or incorrect order of the event according to the model. Thus, semantically the *false* value indicates that the event is aligned with the partial model, and the *true* value indicates otherwise. These variables are the key to obtain the log and model moves in the alignment calculation as will be seen in the objective function. These variables are also used to enable/disable the firing of the reified constraints of the COP.
- *Constraints to enforce Log Traces*: According to the log-relation of the events in the trace, the events are enforced to take those positions. Thus, a set of reified constraints are built to represent conditions of the position of the events

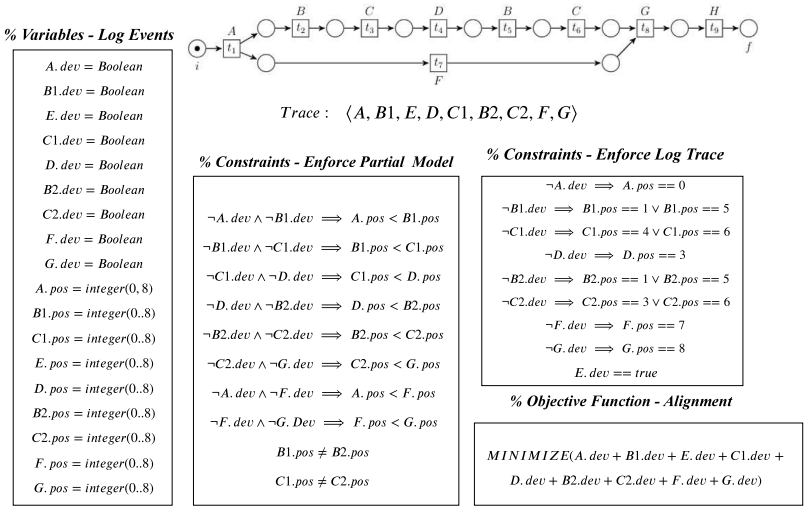


Fig. 13. COP for the example of Fig. 6.

with respect to the log trace. For instance, event *A* occurs first:

$$\neg A.dev \implies A.pos == 0 \tag{1}$$

In the case where the event does not occur in the partial model, it is a deviation, and therefore a constraint is included to force the establishment of a *true* value for the *dev* variable of the event, as occurs with event *E*:

$$E.dev == true \tag{2}$$

In the case of the repeated events, the COP must evaluate all the possibilities of occurrence, as in the case of *B1* and *B2*. The reified constraint must consider the two possibilities, as follows:

$$\neg B1.dev \implies B1.pos == 1 \vee B1.pos == 5 \tag{3}$$

- *Constraints to Enforce Partial Model Run*: These reified constraints represent conditions of the position (*pos*) of the events with regard to the partial model. The reified constraint describes whether an event can be aligned according to the partial model. According to the flow-relation of the partial model, we build reified constraints to represent the related 'later than'-relations between the occurrences of transitions. It should be taken into account that, in the partial models used in our proposal, the XORs are eliminated, and every transition of the model participates in any correct event log. Therefore, the next constraint is an example of this type of reified constraint:

$$\neg A.dev \wedge \neg B1.dev \implies A.pos < B1.pos \tag{4}$$

The reified constraint means that if events *A* and *B1* are aligned with the model, then the value assigned to *pos* of event *A* has to be lower than the values of *pos* of event *B1*. In the case of repeated events (e.g., *B1* and *B2*), extra constraints have to be included in order to prevent their occurrence at the same position:

$$B1.pos \neq B2.pos \tag{5}$$

When a transition in the model is not supported by the execution of an event (taking into account that in the partial model supported by the proposal every transition must be involved in a correct trace since only *and*-branches are included), constraints related to this transition are not added, although a misalignment will be included (a model move). See below for a description of how this is computed.

- *Optimisation function*: The objective function strives to find a solution that minimises the number of deviations. The Boolean variables are considered as Integer, that is, *false* is the 0 value and *true* is the 1 value. As shown in Fig. 13, the objective function is the minimisation of the sum of all deviation variables of our problem. Thus, finding a solution (an assignment) where all the *dev* variables are fixed as *false*, means that every event of the log trace is aligned with the partial model. In the case where any *dev* variable is fixed to *true*, the alignment will be, at least, the number of *dev true* values.

This COP enables the possible deviations between the partial models and the events to be detected:

- **Log moves:** The log moves are determined by consulting the *false* values fixed in the deviation (*dev*). If the *dev* variable of an event reach a *false* value, then this event does not produce a log move. When an event does not occur in the partial model, this situation is a log move, and therefore this situation is controlled by forcing the *true* value in the *dev* variable of the event.
- **Model moves:** Model moves occur when there exists a transition in the partial model that does not occur in the log trace. This situation is easy to identify since a partial model is conflict-free (see Definition 7), meaning that all the transitions must occur in a partial model run. Hence, this situation is penalised as a model move by adding one to the alignment cost function.

Subsequent to the COP resolution, the log and model moves are known, and therefore the alignment cost function can be determined as follows:

$$alignment = \underbrace{\sum_{e_i \in Tr}^{i} e_i.dev}_{\text{log moves}} + \underbrace{\sum_{e_i \in Pm \wedge e_i \notin Tr}^{i} 1}_{\text{model moves}} \tag{6}$$

For the example, the COP reached two optimal solutions where the alignment is equal to 3: one value for the $E.dev = true$, and another value due to the $C1.dev = true$ ($D.dev = true$ in the other equivalent solution) since it is impossible to locate it according to the log trace, and another value because H does not occur in the log trace.

The inclusion of the *time limit* is crucial in Constraint Programming since the solvers return partial solutions during its execution. If the solver is stopped by the time limit, then we have, at least, the best option found up to that moment, although it may not be the global optimal since the search space has not been completely solved.

6. Experiments and evaluation

In order to evaluate our proposal, we have performed various tests to compare the distributed approach proposed in this paper with the classic standalone A* algorithm for computing alignments. Regarding the distributed approach, two algorithms are employed for encoding alignments: the A* algorithm and the COP-based approach presented in this paper (see Section 5). This section is structured as follows:

- Design of the architecture and the technology stack to support our framework (see Section 6.1).
- Selection of a set of representative datasets (see Section 6.2) that includes examples which work better with the distributed approach proposed in this paper, and examples which work better with the classic standalone approach. The configuration of the parameters is also studied.
- Analysis of the approach proposed in this paper, by solving the alignment subproblems derived from the previously selected datasets. The distributed approach is compared to the classic standalone approach (see Section 6.3). For the evaluation and comparison, the performance is considered in terms of the Elapsed Real Time (*ERT*)[7] related to the computation of the alignments.

6.1. Architecture

We propose the use of a three-layer architecture, as shown in Fig. 14. Additionally, we include information regarding the technological stack that has been employed to instantiate this architecture and to perform the experiments.

- *Storage Layer*. The role of this layer is to store the log and process model so that these can be accessed by the nodes that comprise the system. In our particular implementation, it is based on Hadoop HDFS,[8] which is a distributed storage system.
- *Persistence Layer*. This layer is intended to store the results of the alignments. Our implementation relies on the NoSQL database MongoDB.[9]
- *Computing layer*. It is intended to perform the computing operations related to the generation and distribution of partitions and computing alignments. Our implementation is based on Apache Spark,[10] which is a distributed computing framework that enables users to implement applications for the distribution of tasks and Big Data processing.

The mechanism for the generation and distribution of partitions explained in Section 4.2 has been implemented in Apache Spark. As mentioned earlier, we have integrated the PM4Py[11] platform for the computation of alignments using the A* algorithm. On the other hand, the COPs have been implemented with ILOG CPLEX[12] although other solvers can be applied. Regarding the architecture of the Apache Spark cluster (i.e., the architecture of the Computing layer), it is composed of five nodes. Each node is configured with 16GB of RAM and 4 CPUs. This cluster is composed of three types of nodes:

- *Cluster manager*. This is responsible for monitoring and assigning resources among the nodes of the cluster. There is one node entirely dedicated to this task.
- *Driver program*. This node is responsible for distributing the tasks among the Executor nodes. Regarding our implementation, it will schedule the partitioning process by assigning the partitions and their related tasks to the executor nodes. In our case, the driver program is configured to use 8 GB of RAM and 1 CPU by default, and it is run on one of the five nodes of the cluster.
- *Executor nodes*. These nodes execute the tasks assigned by the driver program. They receive the partitions and execute their corresponding tasks. We configured each executor node with 8 GB of RAM and 4 CPUs by default. Each of the four nodes of the cluster hosts one executor.

Both the source code with the implementation of the framework and the datasets used for the experimentation are available at http://www.idea.us.es/confcheckingbigdata/.

6.2. Setting experiments

Five benchmark datasets have been used for the experiments. These are composed of a set of files in XES format as event logs and a set of partial models in Labelled partial order (LPO) format [37]. For a better understanding, the LPO format is a simplification but remains compatible with the PNML format. An LPO represents a run of a place/transition Petri net if it is enabled w.r.t the net. The events of the LPO modelling transition occurrences can fire in the net in accordance with the concurrency and dependency relations given by the LPO.

The event logs and Petri nets employed to illustrate how our proposal works with problems of different sizes are extracted from [20,38,39], whereas the partial models are obtained from the unfolding of Petri nets. Table 1 summarises the dimensions of the datasets employed for this evaluation in terms of: (1) the event logs (number of cases, events, variants, and size); (2) Petri net (number of places, transitions, arcs, and the Cardoso metric (CFC) [40]); and (3) partial models (number of unfolded partial models, number of problems to compute to solve the alignment problem and size, regarding the number of problems to tackle).

Once traces and partial models are combined, the total number of subproblems (see Num. of Problems of Table 1) is derived from the application of the Cartesian product and their required theoretical storage space. The objective of this table involves obtaining an estimate of the complexity involved in the solution of all the problems of each dataset, especially *M5* and *prGm6*, in which more than five and forty million subproblems must be solved, respectively. Approximately, the approach must manage a total of 96 GB and 2.5 TB of data volume, as appear for *M5* and *prGm6* in the table. The distribution of the alignment computation

[7] The *Elapsed Real Time* (*ERT*) is the time from the start of the execution of a program to its completion.

[8] HDFS: https://hadoop.apache.org/.

[9] MongoDB: https://www.mongodb.com/.

[10] Apache Spark: https://spark.apache.org/.

[11] PM4Py: https://pm4py.fit.fraunhofer.de/.

[12] IBM-ILOG CPLEX: https://www.ibm.com/products/ilog-cplex-optimization-studio.

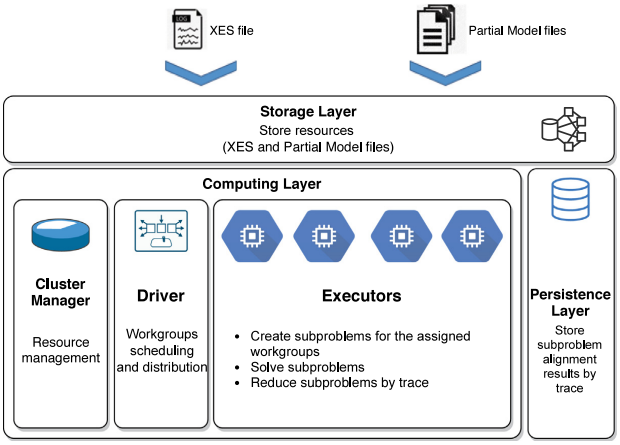


Fig. 14. Cluster architecture.

can imply the transfer of a vast amount of data between the executors, thereby producing a negative impact on the performance. For this reason, the access to traces and partial models has been centralised and they are each accessed by a unique identifier.

The main aspects that might influence the performance of our approach include the setup configuration in terms of the number of partitions for the set of traces (n), and the number of partitions for a set of partial models (m). It is crucial to ascertain out the best setup in terms of the timeout, n, and m, albeit dependent on the type of problem. These parameters are configured as explained below:

- **Grouping the subproblems from the same trace in a partition helps to reduce the number of subproblems to solve**. The approach is optimised in order to prevent the execution of subproblems for two reasons: *(a)* a subproblem is executed iff its estimation of the alignment is lower than the best alignment value obtained for the subproblem related to the same trace in the same partition; and *(b)* the subproblems are sorted by estimation in ascending order. In consequence, when any subproblem is not executed for the aforementioned reason, the execution of the remaining subproblems related to the same trace are skipped since the estimation will always be worse. For this reason, a good setup should concentrate all the problems related to the same trace in the fewest number of partitions as possible, but without over-reducing the parallelisation. By setting n equal to the number of cases in the log, we can assure that each partition will contain subproblems related to the same trace. Hence, this parameter is set to: 500 for $M2$, $M5$ and $M8$; 1265 for $CCC20d$, and 1200 for $prGm6$.
- **Balancing the number of partitions of alignment subproblems**. Some alignment subproblems are too complex and may lead to the formation of bottlenecks in the resolution of the whole alignment problem. A proper partitioning might help in preventing such bottlenecks. For instance, if there is a low number of partitions, it is possible to take advantage of the factor explained above (i.e., avoiding mixing subproblems generated from different traces). The drawback of a low number of partitions is that there could be one or several partitions with a group of complex subproblems which would disproportionately increase the workload of certain executors, while simultaneously leaving other executors idle. The other extreme involves having a high number of partitions. In such a case, the number of subproblems to be solved would increase, since it would not take advantage of the factor previously explained. In this situation, we assume the rule of thumb to be that the higher the number of partitions is, the more subproblems are solved. However, the lower the number of partitions is, the fewer subproblems are solved, although this situation may trigger the creation of bottlenecks. Since it is not possible to know what is the best number of partitions for each dataset, we will test the following values for m in the tests: 1, 2, 4, 5, 6, 8, 12, and 16.

In addition, for each configuration, 10 executions will be performed, and all the results depicted are the average of those executions.

6.3. Results of the experiments

This section is organised as follows. First, Section 6.3.1 highlights the results obtained from the distributed approach proposed in Section 4. Two algorithms are compared with this approach: the A* algorithm, and the COP-based approach. Secondly, Section 6.3.2 compares those results with the classic standalone approach with the A* algorithm.

The metric we employ in order to measure and compare each approach is the *Elapsed Real Time* (*ERT*)[13] for each scenario and dataset. Each *ERT* shown in this section comprises the average of ten executions.

The evaluation is performed by means of an analytic study. This study includes: (i) graphics which depict how the *ERT* evolves as the number of partitions increases; and (ii) an analysis of the trendline of each dataset. By analysing the slope of each trendline, we can quantitatively observe how the *ERT* scales as the number of partitions increases. For each dataset, if the slope is positive, it means that the creation of more partitions (i.e., a better distribution) is not beneficial for that dataset, and the greater the slope is, the worse it scales. If the slope is negative, it means that the approach does scale well for that dataset (and the less steep the slope is, the better it scales). The slope also enables a comparison between several datasets with the same configuration.

[13] The *Elapsed Real Time* (*ERT*) is the time from the start of the execution of a program to the end of it.

Table 1
Datasets used for the experimentation.

Dataset	Event log				Petri net				Partial models		
	Cases	Events	Variants	Size (MB)	Places	Transitions	Arcs	CFC	Num. of models	Num. of problems	Size (MB)
*M*2 [20]	500	8,809	500	2.20	34	34	160	36	102	51,000	509.4
*M*5 [20]	500	17,028	500	4.2	35	33	156	35	10,545	5,272,500	96,989
*M*8 [20]	500	8,246	432	2.1	17	15	72	18	4,1590	2,079.5	31,408.733
*CCC*20*d* [38]	1265	28,440	732	13.3	45	44	94	47	26	32,890	346.619
*prGm*6 [39]	1200	171,685	335	41.8	714	335	1644	383	33,457	40,148,400	2,488,254.81

6.3.1. Results obtained from the distributed approach

The chart in Fig. 15a shows the evolution of the *ERT* increasing the number of partitions of the set of partial models (m) for the datasets *M*2, *M*5, *M*8, and *CCC*20*d*. For these tests, no timeout has been established since the A* algorithm can solve all the subproblems in a reasonable time. Therefore, all the alignments that have been obtained are optimal. Note that the results for *prGm*6 are not in the chart because of the complexity in that particular case, and it is therefore analysed separately.

In detail, the best *ERTs* for *M*5 and *M*8 have been obtained with $m = 2$. From there, the *ERTs* tend to worsen. If the slopes of their trendlines are analysed, that for *M*5 is 0.08, while that for *M*8 is 0.16. For *M*2 and *CCC*20*d*, the best *ERT* is obtained with $m = 1$, and the slopes of their trendlines are 0.22 and 0.57, respectively. This means that *M*5 benefits more from the distribution than the other three datasets.

As aforementioned, the results obtained for the dataset *prGm*6 are depicted in Fig. 15b. Due to memory issues arising from the size of the partitions, it has not been possible to employ values for m from 1 to 6. For this reason, we have used the following values for m in this benchmark: 8, 10, 12, 13, 14, 16, 20, and 24. The best *ERT* value is obtained for $m = 24$, with the slope of the trendline at -1.28. It means that this dataset benefits from a better distribution, since the *ERT* tends to decrease as the number of partitions increases.

From these results, we can conclude that the datasets that produce a larger number of subproblems more benefit are from a larger distribution. It is especially noticeable in the *prGm*6 dataset, as it has a clear tendency to decrease the *ERT* when the number of partitions increases. It should be noted that *M*5, which also produces a large number of alignment subproblems, has a slope of 0.08, which shows better scalability than *M*8, *M*2, and *CCC*20*d*.

Next, we present the results obtained from the distributed approach with the COP-based approach. Fig. 15c shows the evolution of the *ERT* increasing the number of partitions of the set of partial models (m). Once again, the larger m becomes, the smaller are the distributed partitions. For these tests, a timeout of 500 ms per subproblem has been established, since the COP is not capable of solving all the subproblems in a reasonable time if it is unbounded (note that the datasets which have a large number of subproblems might contain a high number thereof producing bottlenecks). Therefore, certain alignments might not achieve the optimal. In the next section, the percentage of traces per dataset for which an optimal alignment value is found will be shown and analysed. Due to the excessive memory consumption by the COPs, it has been impossible to successfully complete any of the executions for the *prGm*6 dataset; hence no results are shown.

Analysing the results for *M*5 and *M*8, the best *ERT* is obtained for $m = 6$. The slopes of the trendline for the two datasets are -3.13 and -1.44, respectively. This shows that the distribution of the alignment subproblems improve the resolution in terms of *ERT*. This is observed when *ERT* is compared to the previous results with the A* algorithm, where the slopes for *M*5 and *M*8 are 0.08 and 0.16, respectively. However, in absolute terms, the *ERT* tends to be higher with the COP solver.

On the other hand, for *M*2 and *CCC*20*d*, the best *ERT* value is obtained for $m = 1$. The slopes of both datasets are 0.02, and 0.06, respectively, which also shows a better scaling than in the previous case with the A* algorithm. For these datasets, the *ERTs* are similar to those obtained with the A* algorithm.

6.3.2. Comparing the A algorithm in standalone with the distributed approaches*

Fig. 16a presents a comparison between the *ERT* of the A* algorithm in standalone, and the best results obtained from the distributed approach.[14,15]

Fig. 16b depicts the percentage of optimal alignments found over the set of traces for each dataset. Note that the only dataset for which the COP-based approach is able to find an optimal value for all the traces is *CCC*20*d*.

In summary, the distributed approach attains better results for *M*2, *M*5, and *prGm*6 with the A* algorithm, and for *CCC*20*d* with the COP-based approach. The datasets which produce a larger number of subproblems, in general, gain greater benefits from a larger distribution (note that the trendlines of *M*5 and *M*8 have a negative slope when solved with the COP-based approach, and a slightly positive tendency when solved with A*). Note also that datasets with extremely complex models might be solved in a reasonable time with the distributed approach, as can be seen with *prGm*6. However, in certain cases, the application of decomposition and the distribution of the subproblems fails to produce the best results. For example, the best results of *M*8 were obtained from the standalone A* algorithm. This is due to the characteristics of the model. An in-depth study into the factors which make a decomposition worthwhile in terms of *ERT* could be performed in the future.

By comparing the distributed approach with A* and the COP-based approach, we can conclude that the more complex the algorithm for computing alignments is, the more benefits the execution attains from a larger distribution. This is justified by the fact that the slopes of the trendlines tend to be closer to zero or negative as the time spent by the subproblems increases. It is especially noticeable in the case of the COP-based approach, which takes more time per subproblem than does the A* algorithm.

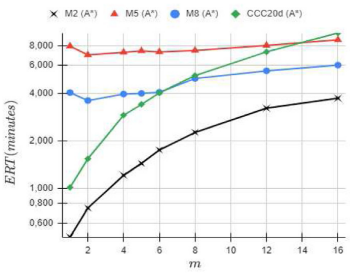
In the light of the results, we can conclude that our approach to decomposing the alignment problem into subproblems and their subsequent distribution, in general, achieves better results in terms of *ERT* in comparison with the standalone approach. Finally, we remark that the complexity of the conformance checking algorithm exerts a heavy influence both on the *ERT* and on the number of optimal alignments (e.g., the COP-based approach).

7. Conclusion

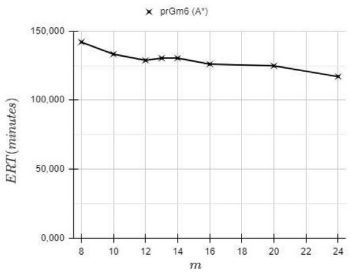
In this paper, a Big Data framework is provided for the parallelisation and distribution of the conformance checking analysis disengaged from the algorithm applied. The creation of subproblems that can be solved distributed makes it possible to tackle

[14] There are no results for the COP-based approach in standalone since the COP implementation proposed in this paper was only conceived to be performed in distributed scenarios.

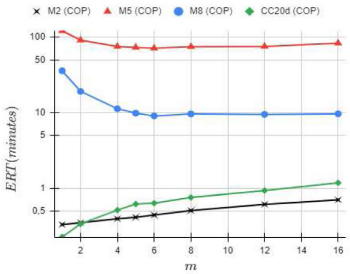
[15] There are no results for the *prGm*6 and the A* algorithm in standalone because the PM4Py execution took more than 24 h without yielding any result.



(a) Benchmark for the datasets *M2*, *M8*, *M5* and *CC20d* by using the A* algorithm distributed.

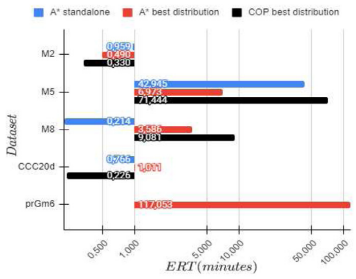


(b) Benchmark for the datasets *prGm6* by using the A* algorithm distributed.

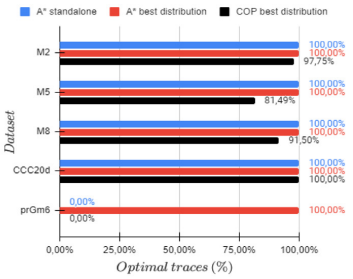


(c) Benchmark for the datasets *M2*, *M8*, *M5* and *CC20d* by using the distributed COP-based approach.

Fig. 15. Results in terms of ERT for the distributed algorithm (logarithmic scale).



(a) Comparison of the best configuration for each algorithm. The abscissa has a logarithmic scale.



(b) Percentage of optimal alignments found over the set of traces for each dataset.

Fig. 16. Comparison between the standalone A* and the distributed A* and the COP-based approach in terms of ERT and percentage of optimal traces.

problems whose complexity could not be approached with local algorithms. For the decomposition, we have proposed an innovative horizontal technique to build subproblems whose resolution is based on a map function, and combined by a *reduceByKey* strategy, with the improvement of an estimation metric that prevents the resolution of unpromising subproblems.

The proposed framework includes the capacity of customising the distribution of models and traces to determine the best configuration for the distribution of the alignment resolution. To demonstrate the applicability of our proposal, the framework has been tested by two alignment techniques: the classic A* approach and a new approach based on the Constraint Optimisation paradigm. The analysis of these two options is derived from the interest in comparing a classic solution with others, such as Constraint Optimisation Problems, which enables the domain to be enclosed and the amount of time available for finding an optimal alignment value to be limited. Five different datasets have been used for testing our framework to compare local (standalone) and distributed solutions, the distributed solution among them, and the effects of the configuration of the distribution on the performance. In summary, the framework provides a high degree of flexibility, since it facilitates the tuning of the parameters that determine the level of distribution of the subproblems, the application of different alignment algorithms, and the applicability of an estimation of the alignment, before it is computed, in order to prevent the analysis of unpromising subproblems.

From this analysis of the experiments, it is possible to find examples where a local solver is more efficient, but for other examples, the distribution of the problem is more efficient than the local. By comparing the two algorithms in distributed scenarios, it is possible to pinpoint the problem areas where each algorithm can find a better solution or take a shorter time. This is why we plan to carry out a more in-depth analysis of the features of the models and logs in order to characterise the problems in terms of ascertaining which performs better before computing the alignments.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work has been partially funded by the Ministry of Science and Technology of Spain ECLIPSE (RTI2018-094283-B-C33) project, the European Regional Development Fund (ERDF/FEDER), MINECO, Spain (TIN2017-86727-C2-1-R), and by the University of Seville, Spain with *VI Plan Propio de Investigación y Transferencia* (VI PPIT-US).

Disclosure

All the authors are responsible for the concept of the paper, the results presented and the writing. All the authors have approved the final content of the manuscript. No potential conflict of interest was reported by the authors.

References

[1] M. Dumas, W.M.P. van der Aalst, A.H.M. ter Hofstede, Process-Aware Information Systems: Bridging People and Software Through Process Technology, Wiley, 2005, URL: http://eu.wiley.com/WileyCDA/WileyTitle/productCd-0471663069.html.

[2] H. Roehm, J. Oehlerking, M. Woehrle, M. Althoff, Model conformance for cyber-physical systems: A survey, TCPS 3 (2019) 30:1–30:26, http://dx.doi.org/10.1145/3306157.

[3] J. Carmona, B.F. van Dongen, A. Solti, M. Weidlich, Conformance Checking - Relating Processes and Models, Springer, 2018, http://dx.doi.org/10.1007/978-3-319-99414-7.

[4] A. Adriansyah, Aligning Observed and Modeled Behavior (Ph.D. thesis), Technische Universiteit Eindhoven, 2014.

[5] W.M.P. van der Aalst, Decomposing Petri nets for process mining: A generic approach, Distrib. Parallel Databases 31 (2013) 471–507, http://dx.doi.org/10.1007/s10619-013-7127-5.

[6] J. Munoz-Gama, J. Carmona, W.M.P. van der Aalst, Single-entry single-exit decomposed conformance checking, Inf. Syst. 46 (2014) 102–122, http://dx.doi.org/10.1016/j.is.2014.04.003.

[7] H.M.W. Verbeek, W.M.P. van der Aalst, Merging alignments for decomposed replay, in: F. Kordon, D. Moldt (Eds.), Application and Theory of Petri Nets and Concurrency: 37th International Conference, PETRI NETS 2016, Toruń, Poland, June 19-24, 2016;
Proceedings, Springer International Publishing, Cham, 2016, pp. 219–239, http://dx.doi.org/10.1007/978-3-319-39086-4_14.

[8] W.L.J. Lee, H.M.W. Verbeek, J. Munoz-Gama, W.M.P. van der Aalst, M. Sepúlveda, Recomposing conformance: Closing the circle on decomposed alignment-based conformance checking in process mining, Inform. Sci. 466 (2018) 55–91, http://dx.doi.org/10.1016/j.ins.2018.07.026.

[9] W.M.P. van der Aalst, Distributed process discovery and conformance checking, in: J. de Lara, A. Zisman (Eds.), Fundamental Approaches to Software Engineering, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 1–25.

[10] J. Dean, S. Ghemawat, MapReduce: Simplified data processing on large clusters, in: E.A. Brewer, P. Chen (Eds.), 6th Symposium on Operating System Design and Implementation (OSDI 2004), USENIX Association, San Francisco, 2004, pp. 137–150, URL: https://ai.google/research/pubs/pub62.

[11] S. Sakr, Z. Maamar, A. Awad, B. Benatallah, W.M.P. van der Aalst, Business process analytics and big data systems: A roadmap to bridge the gap, IEEE Access 6 (2018) 77308–77320, http://dx.doi.org/10.1109/ACCESS.2018.2881759.

[12] M.T. Gómez-López, D. Borrego, J. Carmona, R.M. Gasca, Computing alignments with constraint programming: The acyclic case, in: Proceedings of the International Workshop on Algorithms & Theories for the Analysis of Event Data 2016 Satellite event of the conferences (ATAED) 2016, Torun, Poland, June 20-21, 2016, 2016, pp. 96–110. URL: http://ceur-ws.org/Vol-1592/paper07.pdf.

[13] B.F. van Dongen, J. Carmona, T. Chatain, F. Taymouri, Aligning modeled and observed behavior: A compromise between computation complexity and quality, in: Advanced Information Systems Engineering - 29th International Conference, Essen, Germany, June 12-16, 2017, 2017, pp. 94–109.

[14] B.F. van Dongen, Efficiently computing alignments - using the extended marking equation, in: Business Process Management - 16th International Conference, BPM 2018, Sydney, NSW, Australia, September 9-14, 2018, Proceedings, 2018, pp. 197–214.

[15] M. de Leoni, A. Marrella, Aligning real process executions and prescriptive process models through automated planning, Expert Syst. Appl. 82 (2017) 162–183.

[16] D. Reißner, R. Conforti, M. Dumas, M.L. Rosa, A. Armas-Cervantes, Scalable conformance checking of business processes, in: OTM CoopIS, Rhodes, Greece, 2017, pp. 607–627.

[17] S.J.J. Leemans, D. Fahland, W.M.P. van der Aalst, Scalable process discovery and conformance checking, Softw. Syst. Model. 17 (2018) 599–631, http://dx.doi.org/10.1007/s10270-016-0545-x.

[18] D. Reißner, A. Armas-Cervantes, R. Conforti, M. Dumas, D. Fahland, M. La Rosa, Scalable alignment of process models and event logs: An approach based on automata and s-components, Inf. Syst. 94 (2020) 101561, http://dx.doi.org/10.1016/j.is.2020.101561, URL: http://www.sciencedirect.com/science/article/pii/S0306437920300545.

[19] F. Taymouri, J. Carmona, A recursive paradigm for aligning observed behavior of large structured process models, in: 14th International Conference of Business Process Management (BPM), Rio de Janeiro, Brazil, September 18-22, 2016, pp. 197–214.

[20] F. Taymouri, J. Carmona, Model and event log reductions to boost the computation of alignments, in: P. Ceravolo, C. Guetl, S. Rinderle-Ma (Eds.), Data-Driven Process Discovery and Analysis, Springer International Publishing, 2018, pp. 1–21.

[21] F. Taymouri, J. Carmona, Computing alignments of well-formed process models using local search, ACM Trans. Softw. Eng. Methodol. 29 (2020) 15:1–15:41, http://dx.doi.org/10.1145/3394056.

[22] A. Burattin, S.J. van Zelst, A. Armas-Cervantes, B.F. van Dongen, J. Carmona, Online conformance checking using behavioural patterns, in: Business Process Management - 16th International Conference, BPM 2018, Sydney, NSW, Australia, September 9-14, 2018, Proceedings, 2018, pp. 250–267.

[23] F. Taymouri, J. Carmona, Structural computation of alignments of business processes over partial orders, in: 19th International Conference on Application of Concurrency to System Design, ACSD 2019, Aachen, Germany, June 23-28, 2019, 2019, pp. 73–81.

[24] L. Padró, J. Carmona, Approximate computation of alignments of business processes through relaxation labelling, in: Business Process Management - 17th International Conference, BPM 2019, Vienna, Austria, September 1-6, 2019, Proceedings, 2019, pp. 250–267.

[25] J. Evermann, Scalable process discovery using map-reduce, IEEE Trans. Serv. Comput. 9 (2016) 469–481, http://dx.doi.org/10.1109/TSC.2014.2367525.

[26] F. Chesani, A. Ciampolini, D. Loreti, P. Mello, Map reduce autoscaling over the cloud with process mining monitoring, in: M. Helfert, D. Ferguson, V. Méndez Muñoz, J. Cardoso (Eds.), Cloud Computing and Services Science, Springer International Publishing, Cham, 2017, pp. 109–130.

[27] J. Engelfriet, Branching processes of petri nets, Acta Inf. 28 (1991) 575–591, http://dx.doi.org/10.1007/BF01463946.

[28] R. Bergenthum, S. Mauser, R. Lorenz, G. Juhás, Unfolding semantics of petri nets based on token flows, Fund. Inform. 94 (2009) 331–360, http://dx.doi.org/10.3233/FI-2009-134.

[29] D. Fahland, W.M.P. van der Aalst, Simplifying discovered process models in a controlled manner, Inf. Syst. 38 (2013) 585–605, http://dx.doi.org/10.1016/j.is.2012.07.004.

[30] R. Dechter, Constraint Processing (The Morgan Kaufmann Series in Artificial Intelligence), Morgan Kaufmann, 2003.

[31] K. Apt, Principles of Constraint Programming, Cambridge University Press, New York, NY, USA, 2003.

[32] M.T. Gómez-López, R. Ceballos, R.M. Gasca, C.D. Valle, Developing a labelled object-relational constraint database architecture for the projection operator, Data Knowl. Eng. 68 (2009) 146–172, http://dx.doi.org/10.1016/j.datak.2008.09.002.

[33] A.J. Varela-Vaca, L. Parody, R.M. Gasca, M.T. Gómez-López, Automatic verification and diagnosis of security risk assessments in business process models, IEEE Access 7 (2019) 26448–26465, http://dx.doi.org/10.1109/ACCESS.2019.2901408.

[34] M.T. Gómez-López, R.M. Gasca, J.M. Pérez-Álvarez, Compliance validation and diagnosis of business data constraints in business processes at runtime, Inf. Syst. 48 (2015) 26–43, http://dx.doi.org/10.1016/j.is.2014.07.007.

[35] M.T. Gómez-López, L. Parody, R.M. Gasca, S. Rinderle-Ma, Prognosing the compliance of declarative business processes using event trace robustness, in: On the Move to Meaningful Internet Systems: OTM 2014 Conferences - Confederated International Conferences: CoopIS, and ODBASE 2014, Amantea, Italy, October 27-31, 2014, Proceedings, 2014, pp. 327–344.

[36] D. Borrego, R. Eshuis, M.T. Gómez-López, R.M. Gasca, Diagnosing correctness of semantic workflow models, Data Knowl. Eng. 87 (2013) 167–184.

[37] R. Bergenthum, S. Mauser, Synthesis of petri nets from infinite partial languages with viptool, in: 15th German Workshop on Algorithms and Tools for Petri Nets, Algorithmen und Werkzeuge für Petrinetze, AWPN 2008, Rostock, Germany, September 26-27, 2008, Proceedings, 2008, pp. 81–86. URL: http://ceur-ws.org/Vol-380/paper13.pdf.

[38] J. Buijs, Environmental permit application process ('wabo'), in: CoSeLoG Project, 2014, http://dx.doi.org/10.4121/UUID:26ABA40D-8B2D-435B-B5AF-6D4BFBD7A270, URL: https://data.4tu.nl/collections/Environmental_permit_application_process_WABO_CoSeLoG_project/5065529.

[39] J. Munoz-Gama, J. Carmona, W.M.P. van der Aalst, Conformance checking in the large: Partitioning and topology, in: F. Daniel, J. Wang, B. Weber (Eds.), Business Process Management, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 130–145.

[40] J. Cardoso, Business process control-flow complexity: Metric, evaluation, and validation, Int. J. Web Serv. Res. 5 (2008) 49–76, URL: http://www.igi-global.com/bookstore/titledetails.aspx?titleid=1079.

Chapter 4

Conclusion and Future Work

This Chapter concludes this thesis by summarising the main solutions that have been proposed to achieve the objectives that were traced. Section 4.1 highlights the conclusions of this thesis. Afterward, Section 4.2 discusses the limitations of the proposals and the possible extensions that could be commited in the future.

4.1 Conclusion

In this thesis, we have presented innovative techniques to improve three key activities in Big Data pipelines. These are Data Preparation, Data Quality assessment, and Data Analysis. The contributions are intended to improve the Big Data pipelines that use optimisation problems as a mechanisms of Data Analysis to support decision-making processes. Specifically, the optimisation algorithms that we have improved are exhaustive, which are able to explore the entire search space and can guarantee optimal solutions. These can become NP-complete, especially if large amounts of data are involved. Although the Big Data paradigm is ideal for performing this type of computation, we found a lack of support for these techniques in Big Data environments. For this reason, one of the three main objectives of this thesis is to develop solutions to enhance the distribution and computation of Constraint Optimisation Problems (COPs) and big complex optimisation problems in distributed environments (OBJ 3).

When building Big Data pipelines to use these techniques, we found two problems regarding Data Preparation and Data Quality. On the one hand, our case studies required the transformation of data with complex nested structures, since the COPs used this type of data as input. In this regard, we found the opportunity to develop Domain-Specific Languages (DSL) to make Data Transformation processes easier (OBJ 1). However, we realised that poor-quality data spoilt the results of the Data Analysis. Therefore, we found the opportunity to develop a methodology to model Data Quality rules and automate the evaluation of data usability, as well as facilitate their repair (OBJ 2).

To summarise, this thesis is focused on:

1. Improving Data Preparation activities in cases where data present complex and nested structures (OBJ 1).

2. Improving the evaluation of the Data Quality activity (OBJ 2).

3. Enhancing the Data Analysis activity by supporting the distribution and computation of Constraint Optimisation and conformance checking problems in Big Data environments (OBJ 3).

With respect to the first objective **(OBJ 1)**, we sought to support the Data Transformation process when complex data structures are involved. As the amount of data available to companies increases, the structure of the data tends to become more complex. However, we found a lack of support for this type of data structures in traditional Data Preparation tools, since these solutions tend to support only table-orientated data. We developed the Data Chameleon framework and DSL. The framework provides the core functionalities, being extensible for different DSLs. On the other hand, Data Chameleon DSL provides a language to make Data Transformation easier. As we posed in our contribution, this DSL allows to perform this type of transformations in a straightforward way, eliminating intermediate abstract operations that table-orientated solutions require. We managed to solve two case studies that required the transformation of data with nested structures. This was run in a distributed scenario, where we studied the scalability of our proposal. We demonstrated that Data Chameleon scales as well as both the size of the dataset and the nested structures increase.

Within the scope of the first objective, we also proposed a DSL to improve the extraction of event logs from semi-structured data: The Event Log Extractor DSL (ELE DSL). This language is intended to facilitate the extraction and transformation of complex data structures to produce an event log following the XES format (remark that this format is a standard to represent event logs and is required by most process mining algorithms). We decided to face this objective since we noticed a lack of proposals to extract and transform event logs from semi-structured data sources, which typically involve complex data structures. In our contribution, we solve a real-world case study from Airbus. We enabled the transformation of the logs from the aircraft manufacturing process, which presented nested data structures.

Regarding the second objective **(OBJ 2)**, we developed a methodology to model Data Quality requirements in a context-aware way. We proposed a hierarchical structure of business rules to (i) validate data attributes; (ii) measure Data Quality dimensions; (iii) assess the Data Quality, and (iv) produce recommendations on the usability of the data. The business rules are modelled by employing the Decision Model and Notation (DMN) standard. DMN allows to graphically implement hierarchies of decision rules. The methodology is supported by a tool (DMN4Spark), that enables to execute decision models specified in a DMN format in Big Data environments. In summary, the solution automates the generation of recommendations on the usability of data. We also produced an extension to this methodology in order to give support to the characterisation of the data usability and the data repair. The cornerstone of this proposal is what we call *usability profiles*, a data structure that enables to group records from the datasets that have similar Data Quality problems. Data repair is based on the corrective actions that can be applied to improve the usability decision associated with a specific usability profile. Our proposal enables the modelling of a catalogue of corrective actions that solve the Data Quality problems that are detected in a usability profile. Then, an algorithm based on constraint optimisation strives to select those corrective actions that minimise the

cost of the data reparation. Both methodologies were tested in a case study that was successfully resolved, allowing, on the one hand, to discard those data records that did not meet the quality levels required in the use case. On the other hand, we were able to detect the most common sets of Data Quality problems within the dataset, and we found the set of corrective actions that entailed the least cost.

Finally, the third objective **(OBJ 3)** is focused on supporting two different types of exhaustive optimisation problem in Big Data scenarios: (i) COPs with distributed data, and (ii) big complex optimisation problems applied to the conformance checking paradigm. We found a lack of support for these techniques, which tend to be NP-complete, presenting scalability issues, especially when large amounts of data are involved. Regarding the COP paradigm, in the literature, the scalability issue has been faced by distributing the search space. This approach is called *Distributed Constraint Optimisation Problem* (DCOP). In our proposal, instead of distributing the whole search space, we sought to distribute individual COPs and to optimise the computation by means of querying operators. In this way, the problems can be computed in groups, generating synergies that enable to bound the search space of the problems within the same group. This approach is called *Constraint Optimisation Problem with Distributed Data* (COPDD). Unlike the DCOPs, the COPDD can be computed by using traditional COP solvers and algorithms. We developed a tool (FABIOLA) to facilitate the computation of COPDDs in Big Data environments. In our contribution, we also solved a real-world case study based on the optimisation of the tariff configuration for the customers of three Spanish electricity companies. We compared the performance of our proposal with traditional non-distributed approaches. While the latest demonstrated linear complexity, our proposal was able to reach under-linear complexity in terms of computation time.

With regard to the computation of big complex optimisation problems, we devised a solution for the particular case of conformance checking techniques, since we noticed a lack of proposals to solve these problems in scenarios with highly complex business processes and distributed data. Therefore, we proposed an innovative technique to decompose complex process models into horizontal runs. This technique is called *horizontal decomposition*. We designed an algorithm to facilitate the combination of these chunks of the process model with the traces of the process execution. The proposal offers parameters that can be tuned in order to optimise the way in which the conformance checking problems are distributed and computed. We developed CC4Spark, a tool that implements this methodology. This allows us to load either process models or decomposed process models and distributed event logs. Users can select different conformance checking algorithms to solve the problems. In the benchmark, we compared the sequential A* algorithm with parallel execution using CC4Spark. In the comparison, we include a novel approach to compute conformance checking problems based on COPs. CC4Spark obtained better results in most of the tests that we performed.

Finally, we would like to highlight that during the development of the artefacts of this thesis, a set of tools has been produced. These have been tested in real-world scenarios in Big Data environments, and can be employed in Big Data pipelines through the data processing framework Apache Spark.

4.2 Future Work

The contributions of this thesis are subject to extensions and improvements in several areas. Next, we specify certain areas of improvement for each of the three stages of the Big Data pipeline on which this thesis is focused. Regarding the **Data Preparation** techniques that we propose, the following areas of improvement can be considered.

- Writing Data Transformation queries can be a complex task, especially for non-expert users. A possible improvement for Data Chameleon could be to automate the transformation of complex data using the by-example paradigm. This paradigm is based on inferring the Data Transformations to be performed from examples provided by the user, so that the user only must indicate an example of a record before and after transforming it. The system would automatically look for the transformations that produce the result indicated by the user.
- Following the previous idea, a possible improvement would be the creation of graphical interfaces to carry out complex transformations. The way of representing the data should be customisable by the user. Therefore, the possibility of establishing different data representation models could be included to facilitate interoperability between different nested levels.

With respect to the **Data Quality** techniques that have been presented in this thesis, the following areas of improvement can be considered.

- DMN4DQ only supports record-by-record Data Quality evaluation. Consequently, it is only capable of detecting Data Quality problems at the record level. For example, DMN4DQ is not able to detect Data Quality problems derived from duplicate data in different records. The methodology could be extended in this direction.
- The methodology is closely related to the DMN in order to model decision rules. A possible line of improvement would be to elevate the level of abstraction of the methodology so that it can be implemented through different business and/or decision rule managers.
- The DMN4DQ+ only enables to repair individual usability profiles. In future developments of the proposal, we contemplate the possibility of repairing more than one usability profile at the same time. This could lead to savings in terms of cost and time.

Regarding the **Data Analysis** techniques that have been proposed related to the computation of exhaustive optimisation problems in distributed environments, we propose the following areas of improvement.

- A possible extension of FABIOLA, our solution to compute COPs in Big Data environments, could be to support optimisation algorithms based on metaheuristics. This would require raising the degree of abstraction of the proposal.

- Another line of improvement for FABIOLA could be to increase the number of queries available to optimise the execution of the COPs. Furthermore, a query catalogue could be formalised along with the available optimisation operations.
- Our solution to computing a big complex optimisation problem is closely tied to the conformance checking paradigm. We believe that it would be interesting to elevate the abstraction level of the proposal and generalise the type of problem that can be solved with the distribution technique that we have proposed.
- Regarding our solution to compute complex conformance checking problems in distributed environments, we could carry out a study on optimal configurations based on the characteristics of the process models and event logs. This would make it easier to configure our proposal based on the type of problem to be faced.
- The *quantum annealing* optimisation technique is a field within quantum computing that has been proven to be capable of solving NP-complete optimisation problems. One possible area of improvement can be the incorporation of quantum annealing in Big Data pipelines.

Bibliography

[1] Laith Abualigah et al. "Advances in Meta-Heuristic Optimization Algorithms in Big Data Text Clustering". In: *Electronics* 10.2 (2021). ISSN: 2079-9292. DOI: 10.3390/electronics10020101.

[2] D. P. Acharjya and Kauser Ahmed. "A Survey on Big Data Analytics: Challenges, Open Research Issues and Tools". In: *International Journal of Advanced Computer Science and Applications* 7.2 (2016). ISSN: 21565570. DOI: 10.14569/IJACSA.2016.070267.

[3] Arya Adriansyah. "Aligning observed and modeled behavior". PhD thesis. Technische Universiteit Eindhoven, 2014.

[4] Ifeyinwa Angela Ajah and Henry Friday Nweke. "Big Data and Business Analytics: Trends, Platforms, Success Factors and Applications". In: *Big Data and Cognitive Computing 2019, Vol. 3, Page 32* 3.2 (2019), p. 32. ISSN: 2504-2289. DOI: 10.3390/BDCC3020032. URL: https://www.mdpi.com/2504-2289/3/2/32/htmhttps://www.mdpi.com/2504-2289/3/2/32.

[5] Adel Alkhalil and Rabie A. Ramadan. "IoT Data Provenance Implementation Challenges". In: *Procedia Computer Science* 109 (2017), pp. 1134–1139. ISSN: 1877-0509. DOI: 10.1016/J.PROCS.2017.05.436. URL: https://www.sciencedirect.com/science/article/pii/S1877050917311183.

[6] Wesley Gongora de Almeida et al. "Taxonomy of data quality problems in multidimensional Data Warehouse models". In: *2013 8th Iberian Conference on Information Systems and Technologies (CISTI)*. 2013, pp. 1–7.

[7] *Apache spark™ - unified engine for large-scale data analytics*. URL: https://spark.apache.org/.

[8] Claudio A. Ardagna, Paolo Ceravolo, and Ernesto Damiani. "Big data analytics as-a-service: Issues and challenges". In: *Proceedings - 2016 IEEE International Conference on Big Data, Big Data 2016*. IEEE, 2016, pp. 3638–3644. ISBN: 9781467390040. DOI: 10.1109/BigData.2016.7841029. URL: http://ieeexplore.ieee.org/document/7841029/.

[9] Claudio A. Ardagna et al. "A Model-Driven Methodology for Big Data Analytics-as-a-Service". In: *Proceedings - 2017 IEEE 6th International Congress on Big Data, BigData Congress 2017*. IEEE, 2017, pp. 105–112. ISBN: 9781538619964. DOI: 10.1109/BigDataCongress.2017.23. URL: http://ieeexplore.ieee.org/document/8029315/.

[10] Danilo Ardagna et al. "Context-aware data quality assessment for big data". In: *Future Generation Computer Systems* 89 (2018), pp. 548–562. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2018.07.014. URL: https://www.sciencedirect.com/science/article/pii/S0167739X17329151.

[11] B. Arputhamary and L. Arockiam. "Data Integration in Big Data Environment". In: *Bonfring International Journal of Data Mining* 5.1 (2015), pp. 01–05. ISSN: 2250107X. DOI: 10.9756/BIJDM.8001. URL: http://journal.bonfring.org/abstract.php?id=2&archiveid=429.

[12] Rolf B. Banziger, Artie Basukoski, and Thierry Chaussalet. "Discovering Business Processes in CRM Systems by Leveraging Unstructured Text Data". In: *2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*. 2018, pp. 1571–1577. DOI: 10.1109/HPCC/SmartCity/DSS.2018.00257.

[13] Cristóbal Barba-González et al. "Injecting domain knowledge in multi-objective optimization problems: A semantic approach". In: *Computer Standards Interfaces* 78 (2021), p. 103546. ISSN: 0920-5489. DOI: 10.1016/J.CSI.2021.103546.

[14] Cristóbal Barba-González et al. "BIGOWL: Knowledge centered Big Data analytics". In: *Expert Systems with Applications* 115 (2019), pp. 543–556. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2018.08.026. URL: https://www.sciencedirect.com/science/article/pii/S0957417418305347.

[15] Carlo Batini and Monica Scannapieco. *Data and information quality: dimensions, principles and techniques*. 2016. ISBN: 9783319241067.

[16] Bloor Research. *Data Preparation (Self-Service)*. 2018. URL: https://www.bloorresearch.com/technology/data-preparation-self-service/.

[17] Christian Blum and Andrea Roli. "Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison". In: *ACM Computing Surveys* 35.3 (2003), 268 – 308. DOI: 10.1145/937503.937505.

[18] J.C.A.M. Buijs. *Environmental permit application process ('WABO'), CoSeLoG project*. nl. 2014. DOI: 10.4121/UUID:26ABA40D-8B2D-435B-B5AF-6D4BFBD7A270. URL: https://data.4tu.nl/collections/Environmental_permit_application_process_WABO_CoSeLoG_project/5065529.

[19] Ismael Caballero et al. "A Data Quality Measurement Information Model Based On ISO/IEC 15939." In: *ICIQ*. Cambridge, MA. 2007, pp. 393–408.

[20] Diego Calvanese et al. "Ontology-Driven Extraction of Event Logs from Relational Databases". In: *Business Process Management Workshops*. Ed. by Manfred Reichert and Hajo A. Reijers. Cham: Springer International Publishing, 2016, pp. 140–153. ISBN: 978-3-319-42887-1.

[21] Josep Carmona et al. *Conformance Checking - Relating Processes and Models*. Springer, 2018. ISBN: 978-3-319-99413-0. DOI: 10.1007/978-3-319-99414-7. URL: https://doi.org/10.1007/978-3-319-99414-7.

[22] Paolo Ceravolo et al. "Big Data Semantics". In: *Journal on Data Semantics* 7.2 (2018), pp. 65–85. ISSN: 18612040. DOI: 10.1007/s13740-018-0086-2. URL: http://link.springer.com/10.1007/s13740-018-0086-2.

[23] Chen Chen et al. "BigGorilla: An Open-Source Ecosystem for Data Preparation and Integration". In: *IEEE Data Eng. Bull.* 41.2 (2018), pp. 10–22.

[24] P. Cong and Z. Xiaoyi. "Research and Design of Interactive Data Transformation and Migration System for Heterogeneous Data Sources". In: *2009 WASE International Conference on Information Engineering*. Vol. 1. 2009, pp. 534–536.

[25] William H. Crown. "The Potential Role of Constrained Optimization Methods in Healthcare Decision Making". In: *Applied Health Economics and Health Policy 2020 18:4* 18.4 (2020), pp. 461–462. ISSN: 1179-1896. DOI: 10.1007/S40258-020-00559-8. URL: https://link.springer.com/article/10.1007/s40258-020-00559-8.

[26] Edward Curry. "The big data value chain: Definitions, concepts, and theoretical approaches". In: *New Horizons for a Data-Driven Economy: A Roadmap for Usage and Exploitation of Big Data in Europe*. Cham: Springer International Publishing, 2016, pp. 29–37. ISBN: 9783319215693. DOI: 10.1007/978-3-319-21569-3_3.

[27] Jeffrey Dean and Sanjay Ghemawat. "MapReduce: Simplified Data Processing on Large Clusters". In: *6th Symposium on Operating System Design and Implementation (OSDI 2004)*. Ed. by Eric A. Brewer and Peter Chen. San Francisco: USENIX Association, 2004, pp. 137–150. URL: https://ai.google/research/pubs/pub62.

[28] Yuri Demchenko et al. "Big security for big data: Addressing security challenges for the big data infrastructure". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 8425 LNCS. Springer, Cham, 2014, pp. 76–94. ISBN: 9783319068107. DOI: 10.1007/978-3-319-06811-4_13. URL: http://link.springer.com/10.1007/978-3-319-06811-4_13.

[29] Xin Luna Dong and Divesh Srivastava. "Big Data Integration". In: *Synthesis Lectures on Data Management* 7.1 (2015), pp. 1–198. ISSN: 2153-5418. DOI: 10.2200/S00578ED1V01Y201404DTM040.

[30] Shaker H. Ali El-Sappagh, Abdeltawab M. Ahmed Hendawi, and Ali Hamed El Bastawissy. "A proposed model for data warehouse ETL processes". In: *Journal of King Saud University - Computer and Information Sciences* 23.2 (2011), pp. 91–104. ISSN: 1319-1578. DOI: 10.1016/J.JKSUCI.2011.05.005. URL: https://www.sciencedirect.com/science/article/pii/S131915781100019X/?imgSel=Y.

[31] Adir Even, G. Shankaranarayanan, and Paul D. Berger. "Evaluating a model for cost-effective data quality management in a real-world CRM setting". In: *Decision Support Systems* 50.1 (2010), pp. 152–163. ISSN: 0167-9236. DOI: https://doi.org/10.1016/j.

dss.2010.07.011. URL: https://www.sciencedirect.com/science/article/pii/S016792361000117X.

[32] André Freitas and Edward Curry. "Big Data Curation". In: *New Horizons for a Data-Driven Economy*. Cham: Springer International Publishing, 2016, pp. 87–118. DOI: 10.1007/978-3-319-21569-3_6.

[33] Eugene C. Freuder and Richard J. Wallace. "Partial constraint satisfaction". In: *Artificial Intelligence* 58.1-3 (1992), pp. 21–70. ISSN: 00043702. DOI: 10.1016/0004-3702(92)90004-H.

[34] Tim Furche et al. "Data Wrangling for Big Data: Challenges and Opportunities". In: *19th International Conference on Extending Database Technology (EDBT)*. Vol. 8062 LNAI. 2016, pp. 473–478. DOI: 10.5441/002/EDBT.2016.44.

[35] Xinbo Geng and Le Xie. "Data-driven Decision Making with Probabilistic Guarantees (Part 1): A Schematic Overview of Chance-constrained Optimization". In: (2019). DOI: 10.48550/arxiv.1903.10621. arXiv: 1903.10621. URL: https://arxiv.org/abs/1903.10621v2.

[36] Boris Glavic. "Big data provenance: Challenges and implications for benchmarking". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 8163 LNCS. Springer Verlag, 2014, pp. 72–80. ISBN: 9783642539732. DOI: 10.1007/978-3-642-53974-9_7. URL: https://link.springer.com/chapter/10.1007/978-3-642-53974-9_7.

[37] Monika Gupta and Ashish Sureka. "Nirikshan: Mining Bug Report History for Discovering Process Maps, Inefficiencies and Inconsistencies". In: *Proceedings of the 7th India Software Engineering Conference*. ISEC '14. Chennai, India: Association for Computing Machinery, 2014. ISBN: 9781450327763. DOI: 10.1145/2590748.2590749. URL: https://doi.org/10.1145/2590748.2590749.

[38] María Teresa Gómez-López and Rafael M. Gasca. "Using Constraint Programming in Selection Operators for Constraint Databases". In: *Expert Systems with Applications* 41.15 (2014), pp. 6773–6785. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2014.04.047. URL: https://www.sciencedirect.com/science/article/pii/S095741741400270X.

[39] *Hadoop HDFS Architecture Guide*. URL: https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html.

[40] Francisco José de Haro-Olmo et al. "Data curation in the Internet of Things: A decision model approach". In: *Computational and Mathematical Methods* 3.6 (2021), e1191. DOI: https://doi.org/10.1002/cmm4.1191. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cmm4.1191. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cmm4.1191.

[41] Joseph M. Hellerstein, Jeffrey Heer, and Sean Kandel. "Self-Service Data Preparation: Research to Practice". In: *IEEE Data Eng. Bull.* 41 (2018), pp. 23–34.

[42] Katsutoshi Hirayama and Makoto Yokoo. "Distributed partial constraint satisfaction problem". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 1330 (1997), pp. 222–236. ISSN: 16113349. DOI: 10.1007/BFB0017442.

[43] Philip Howard. *Data Preparation (self-service) – Bloor Research*. 2018. URL: https://www.bloorresearch.com/technology/data-preparation-self-service/ (visited on 06/14/2022).

[44] "IEEE Standard for eXtensible Event Stream (XES) for Achieving Interoperability in Event Logs and Event Streams". In: *IEEE Std 1849-2016* (2016), pp. 1–50. DOI: 10.1109/IEEESTD.2016.7740858.

[45] Ihab F Ilyas and Xu Chu. *Data cleaning*. ACM, 2019.

[46] Illinois School of Information Sciences. *Foundations of Data Curation*. 2018. URL: https://ischool.illinois.edu/degrees-programs/courses/is531 (visited on 06/14/2022).

[47] *Introduction to MongoDB - MongoDB Manual*. URL: https://www.mongodb.com/docs/manual/introduction/.

[48] Jagreet Kaur. *Data Preprocessing and Data Wrangling Machine Learning*. 2017. URL: https://www.xenonstack.com/blog/data-preparation (visited on 06/14/2022).

[49] Zhongjun Jin et al. "Foofah: Transforming Data By Example". In: *Proceedings of the 2017 ACM International Conference on Management of Data - SIGMOD '17*. New York, New York, USA: ACM Press, 2017, pp. 683–698. ISBN: 9781450341974. DOI: 10.1145/3035918.3064034. URL: http://dl.acm.org/citation.cfm?doid=3035918.3064034.

[50] Jeffrey Heer Joseph M. Hellerstein and Sean Kandel. *Self-Service Data Preparation: Research to Practice*. Tech. rep. Bulletin of the IEEE Computer Society Technical Committee on Data Engineering, 2018.

[51] Nikunj Joshi and Bintu Kadhiwala. "Big data security and privacy issues — A survey". In: *2017 Innovations in Power and Advanced Computing Technologies (i-PACT)*. 2017, pp. 1–5. DOI: 10.1109/IPACT.2017.8245064.

[52] Doug Laney. "3D data management: Controlling data volume, velocity, and variety". In: *META Group* (2001). URL: https://blogs.gartner.com/doug-laney/files/2012/01/ad949-3D-Data-Management-Controlling-Data-Volume-Velocity-and-Variety.pdf.

[53] In Lee. "Big data: Dimensions, evolution, impacts, and challenges". In: *Business Horizons* 60.3 (2017), pp. 293–303. ISSN: 0007-6813. DOI: 10.1016/J.BUSHOR.2017.01.004. URL: https://www.sciencedirect.com/science/article/pii/S0007681317300046.

[54] James Manyika et al. *Big data: The next frontier for innovation, competition, and productivity | McKinsey*. 2011. URL: https://www.mckinsey.com/business-functions/digital-mckinsey/our-insights/big-data-the-next-frontier-for-innovation (visited on 04/25/2022).

[55] Mohsen Marjani et al. "Big IoT Data Analytics: Architecture, Opportunities, and Open Research Challenges". In: *IEEE Access* 5 (2017), pp. 5247–5261. ISSN: 21693536. DOI: 10.1109/ACCESS.2017.2689040. arXiv: 2017. URL: http://ieeexplore.ieee.org/document/7888916/.

[56] Aiswarya Raj Munappy, Jan Bosch, and Helena Homström Olsson. "Data Pipeline Management in Practice: Challenges and Opportunities". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 12562 LNCS (2020), pp. 168–184. ISSN: 16113349. DOI: 10.1007/978-3-030-64148-1_11. URL: https://link.springer.com/chapter/10.1007/978-3-030-64148-1_11.

[57] Eduardo González López de Murillas, Wil M. P. van der Aalst, and Hajo A. Reijers. "Process Mining on Databases: Unearthing Historical Data from Redo Logs". In: *Business Process Management*. Ed. by Hamid Reza Motahari-Nezhad, Jan Recker, and Matthias Weidlich. Cham: Springer International Publishing, 2015, pp. 367–385. ISBN: 978-3-319-23063-4.

[58] Marek Obitko, Václav Jirkovský, and Jan Bezdíček. "Big data challenges in industrial automation". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 8062 LNAI. 2013, pp. 305–316. ISBN: 9783642400896. DOI: 10.1007/978-3-642-40090-2_27.

[59] Paulo Oliveira, Fátima Rodrigues, and Pedro Rangel Henriques. "A Formal Definition of Data Quality Problems." In: *ICIQ*. 2005.

[60] OMG. *Decision Model and Notation (DMN), Version 1.2*. Object Management Group, 2019. URL: https://www.omg.org/spec/DMN.

[61] *Operational support and enterprise systems*. URL: https://www.britannica.com/topic/information-system/Management-support.

[62] Barry O'Sullivan. "Opportunities and Challenges for Constraint Programming". In: *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence*. AAAI'12. Toronto, Ontario, Canada: AAAI Press, 2012, pp. 2148–2152. URL: http://dl.acm.org/citation.cfm?id=2900929.2901033.

[63] Ahmed Oussous et al. "Big Data technologies: A survey". In: *Journal of King Saud University - Computer and Information Sciences* 30.4 (2018), pp. 431–448. ISSN: 1319-1578. DOI: 10.1016/J.JKSUCI.2017.06.001. URL: https://www.sciencedirect.com/science/article/pii/S1319157817300034.

[64] Pekka Pääkkönen and Daniel Pakkala. "Reference Architecture and Classification of Technologies, Products and Services for Big Data Systems". In: *Big Data Research* 2.4 (2015), pp. 166–186. ISSN: 2214-5796. DOI: 10.1016/J.BDR.2015.01.001. URL: https://www.sciencedirect.com/science/article/pii/S2214579615000027.

[65] Ken Peffers et al. "A Design Science Research Methodology for Information Systems Research". In: *https://doi.org/10.2753/MIS0742-1222240302* 24.3 (2014), pp. 45–77. ISSN: 07421222. DOI: 10.2753/MIS0742-1222240302. URL: https://www.tandfonline.com/doi/abs/10.2753/MIS0742-1222240302.

[66] *PM4Py - Process Mining for Python*. Fraunhofer Institute for Applied Information Technology. 2022. URL: https://pm4py.fit.fraunhofer.de/docs.

[67] Shakthi Poornima and Mullur Pushpalatha. "A journey from big data towards prescriptive analytics". In: *ARPN Journal of Engineering and Applied Sciences* 11.19 (2016), pp. 11465–11474. ISSN: 18196608.

[68] Charles Prud'homme, Jean-Guillaume Fages, and Xavier Lorca. *Choco Documentation*. TASC - LS2N CNRS UMR 6241, COSLING S.A.S. 2017. URL: http://www.choco-solver.org.

[69] José Miguel Pérez-Álvarez et al. "Tactical Business-Process-Decision Support based on KPIs Monitoring and Validation". In: *Computers in Industry* 102 (2018), pp. 23–39. ISSN: 0166-3615. DOI: https://doi.org/10.1016/j.compind.2018.08.001. URL: https://www.sciencedirect.com/science/article/pii/S0166361517307819.

[70] Erhard Rahm and Hong Hai Do. "Data cleaning: Problems and current approaches". In: *IEEE Data Eng. Bull.* 23.4 (2000), pp. 3–13.

[71] Edwin D. Reilly, Anthony. Ralston, and David. Hemmendinger. "Backus-Naur form (BNF)". In: *Encyclopedia of Computer Science*. Wiley, 2003, pp. 129–131. ISBN: 0470864125. URL: https://dl.acm.org/citation.cfm?id=1074155.

[72] Dirk Riehle et al. "Composite design patterns". In: *ACM SIGPLAN Notices* 32.10 (1997), pp. 218–228. ISSN: 03621340. DOI: 10.1145/263700.263739. URL: http://portal.acm.org/citation.cfm?doid=263700.263739.

[73] F. Rossi, P. van Beek, and T. Walsh. *Handbook of Constraint Programming*. Elsevier, 2006. ISBN: 978-0-444-52726-4.

[74] Kewei Sha and Sherali Zeadally. "Data Quality Challenges in Cyber-Physical Systems". In: *J. Data and Information Quality* 6.2–3 (2015). ISSN: 1936-1955. DOI: 10.1145/2740965. URL: https://doi.org/10.1145/2740965.

[75] *Software engineering — Software product Quality Requirements and Evaluation (SQuaRE) — Data quality model*. Standard. International Organization for Standardization, 2019.

[76] Jerzy Stefanowski, Krzysztof Krawiec, and Robert Wrembel. "Exploring complex and big data". In: *International Journal of Applied Mathematics and Computer Science* 27.4 (2017), pp. 669–679. ISSN: 2083-8492. DOI: 10.1515/amcs-2017-0046. URL: http://content.sciendo.com/view/journals/amcs/27/4/article-p669.xml.

[77] J Gerald Suarez. *Three Experts on Quality Management: Philip B. Crosby, W. Edwards Deming, Joseph M. Juran*. Tech. rep. TOTAL QUALITY LEADERSHIP OFFICE ARLINGTON VA, 1992.

[78] Ikbal Taleb et al. "Big Data Quality: A Quality Dimensions Evaluation". In: *2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*. IEEE, 2016, pp. 759–765. ISBN: 978-1-5090-2771-2. DOI: 10.1109/UIC-ATC-ScalCom-CBDCom-IoP-SmartWorld.2016.0122. URL: http://ieeexplore.ieee.org/document/7816918/.

[79] *The Practitioner's Guide to Data Quality Improvement*. Elsevier, 2011. DOI: 10.1016/c2009-0-17212-4. URL: https://doi.org/10.1016/c2009-0-17212-4.

[80] *The Scala Programming Language*. URL: https://www.scala-lang.org/.

[81] Deborah L. Thurston and Suresh Srinivasan. "Constrained Optimization for Green Engineering Decision-Making". In: *Environmental Science and Technology* 37.23 (2003), pp. 5389–5397. ISSN: 0013936X. DOI: 10.1021/es0344359. URL: https://pubs.acs.org/doi/full/10.1021/es0344359.

[82] Álvaro Valencia-Parra et al. "CC4Spark: Distributing Event Logs and big complex Conformance Checking problems". In: *Proceedings of the Best Dissertation Award, Doctoral Consortium, and Demonstration and Resources Track at BPM 2021 co-located with 19th International Conference on Business Process Management (BPM 2021), Rome, Italy, September 6th - to - 10th, 2021*. CEUR-WS.org, 2021, pp. 136–140. URL: http://ceur-ws.org.

[83] Álvaro Valencia-Parra et al. "CHAMALEON: Framework to improve Data Wrangling with Complex Data". In: *Proceedings of the 40th International Conference on Information Systems, ICIS 2019, Munich, Germany, December 15-18, 2019*. 2019. URL: https://aisel.aisnet.org/icis2019/data_science/data_science/16.

[84] Álvaro Valencia-Parra et al. "DMN for Data Quality Measurement and Assessment". In: *Business Process Management Workshops*. Ed. by Chiara Di Francescomarino, Remco Dijkman, and Uwe Zdun. Cham: Springer International Publishing, 2019, pp. 362–374. ISBN: 978-3-030-37453-2.

[85] Álvaro Valencia-Parra et al. "Empowering conformance checking using Big Data through horizontal decomposition". In: *Information Systems* 99 (2021), p. 101731. ISSN: 03064379. DOI: 10.1016/j.is.2021.101731. URL: https://linkinghub.elsevier.com/retrieve/pii/S0306437921000077.

[86] Álvaro Valencia-Parra et al. "Enabling Process Mining in Airbus Manufacturing". In: *Business Process Management Cases Vol. 2* (2021), pp. 125–138. DOI: 10.1007/978-3-662-63047-1_10. URL: https://link.springer.com/chapter/10.1007/978-3-662-63047-1_10.

[87] Álvaro Valencia-Parra et al. "Enabling Process Mining in Aircraft Manufactures: Extracting Event Logs and Discovering Processes from Complex Data". In: *Proceedings of the Industry Forum at BPM 2019*. Vienna, 2019, pp. 166–177.

[88] Álvaro Valencia-Parra. "Analysis of Big Data Architectures and Pipelines: Challenges and Opportunities". Master Thesis. Universidad de Sevilla, 2019.

[89] Álvaro Valencia-Parra et al. "DMN4DQ: When data quality meets DMN". In: *Decision Support Systems* 141 (2021), p. 113450. ISSN: 0167-9236. DOI: https://doi.org/10.1016/j.dss.2020.113450. URL: https://www.sciencedirect.com/science/article/pii/S0167923620302050.

[90] Álvaro Valencia-Parra et al. "Unleashing Constraint Optimisation Problem solving in Big Data environments". In: *Journal of Computational Science* 45 (2020), p. 101180. ISSN: 1877-7503. DOI: https://doi.org/10.1016/j.jocs.2020.101180. URL: https://www.sciencedirect.com/science/article/pii/S1877750320304816.

[91] Jianwu Wang et al. "Big data provenance: Challenges, state of the art and opportunities". In: *2015 IEEE International Conference on Big Data (Big Data)*. Vol. 2015. IEEE, 2015, pp. 2509–2516. ISBN: 978-1-4799-9926-2. DOI: 10.1109/BigData.2015.7364047. URL: http://www.ncbi.nlm.nih.gov/pubmed/29399671http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC5796788http://ieeexplore.ieee.org/document/7364047/.

[92] Philip Woodall, Martin Oberhofer, and Alexander Borek. "A classification of data quality assessment and improvement methods". In: *International Journal of Information Quality* 3.4 (2014), pp. 298–321.

[93] Ibrar Yaqoob et al. "Big data: From beginning to future". In: *International Journal of Information Management* 36.6 (2016), pp. 1231–1247. ISSN: 02684012. DOI: 10.1016/j.ijinfomgt.2016.07.009.

[94] Yutong Zhang et al. "A multi-agent genetic algorithm for big optimization problems". In: *2015 IEEE Congress on Evolutionary Computation (CEC)*. 2015, pp. 703–707. DOI: 10.1109/CEC.2015.7256959.

[95] Keliang Zhou, Taigang Liu, and Lifeng Zhou. "Industry 4.0: Towards future industrial opportunities and challenges". In: *2015 12th International Conference on Fuzzy Systems and Knowledge Discovery, FSKD 2015*. IEEE, 2016, pp. 2147–2152. ISBN: 9781467376822. DOI: 10.1109/FSKD.2015.7382284. arXiv: arXiv:1011.1669v3. URL: http://ieeexplore.ieee.org/document/7382284/.