A Multi-functional Provenance Architecture: Challenges and Solutions

A Thesis Submitted to the

College of Graduate Studies and Research

in Partial Fulfillment of the Requirements

for the degree of Doctor of Philosophy

in the Department of Computer Science

University of Saskatchewan

By

Mahsa Naseri

Permission to Use

In presenting this thesis in partial fulfillment of the requirements for a Postgraduate degree from the University of Saskatchewan, I agree that the Libraries of this University may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis work or, in their absence, by the Head of the Department or the Dean of the College in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of Saskatchewan in any scholarly use which may be made of any material in my thesis. Requests for permission to copy or to make other use of material in this thesis in whole or part should be addressed to:

Head of the Department of Computer Science
176 Thorvaldson Building
110 Science Place
University of Saskatchewan
Saskatoon, Saskatchewan
Canada
S7N 5C9

ABSTRACT

In service-oriented environments, services are put together in the form of a workflow with the aim of distributed problem solving. Capturing the execution details of the services' transformations is a significant advantage of using workflows. These execution details, referred to as provenance information, are usually traced automatically and stored in provenance stores. Provenance data contains the data recorded by a workflow engine during a workflow execution. It identifies what data is passed between services, which services are involved, and how results are eventually generated for particular sets of input values. Provenance information is of great importance and has found its way through areas in computer science such as: Bioinformatics, database, social, sensor networks, etc.

Current exploitation and application of provenance data is very limited as provenance systems started being developed for specific applications. Thus, applying learning and knowledge discovery methods to provenance data can provide rich and useful information on workflows and services. Therefore, in this work, the challenges with workflows and services are studied to discover the possibilities and benefits of providing solutions by using provenance data.

A multifunctional architecture is presented which addresses the workflow and service issues by exploiting provenance data. These challenges include workflow composition, abstract workflow selection, refinement, evaluation, and graph model extraction. The specific contribution of the proposed architecture is its novelty in providing a basis for taking advantage of the previous execution details of services and workflows along with artificial intelligence and knowledge management techniques to resolve the major challenges regarding workflows. The presented architecture is application-independent and could be deployed in any area.

The requirements for such an architecture along with its building components are discussed. Furthermore, the responsibility of the components, related works and the implementation details of the architecture along with each component are presented.

ACKNOWLEDGEMENTS

I would like to thank my supervisor, Dr. Simone Ludwig, for giving me an opportunity to pursue a doctorate degree at the university of Saskatchewan. Her encouragement, guidance and support from the initial to the final steps of my PhD studies enabled me to develop an understanding of the research subject and improve my scientific problem solving skills. Particularly, I value her pursuit and long distance supervision as well as her patience and understanding. She has made me competent in organizing and managing research activities such as project proposals, international conferences, and software demonstrations. She has been both a professor and a friend for me within these years.

I would also like to greatly thank my committee members whose comments, and guidelines helped me a lot through my research. I strongly appreciate professor Michael Horsch for all I have learned from his knowledge representation and reasoning class as well as professors Nathaniel Osgood, Gordon McCalla, and Longhai Li for leading me through my PhD. studies, my comprehensive exam, and proposal defense. Their directions and feedbacks provided me with valuable progress in my research. I strongly appreciate the time they invested in reading my thesis document and the effort they made in providing me with their valuable comments.

I would like to devote this thesis to my beloved ones, my parents, Mahroo and Bagher, my sister, Neda, and my brother, Mahdi, whose support, love, and prayers have been with me always.

CONTENTS

Permission to Use i

Abstract ii

Acknowledgements iv

Contents vi

List of Tables ix

List of Figures x

List of Abbreviations xi

1 Introduction 1

1.1 Introduction . . . 1

1.1.1 Service Oriented Architecture . . . 1

1.1.2 Workflow . . . 3

1.1.3 SOA and Provenance . . . 4

1.2 Provenance: Definition, Systems, and Applications . . . 5

1.2.1 Provenance: Definition and Examples . . . 6

1.2.2 Provenance Application Areas . . . 10

1.3 Provenance Systems . . . 15

1.3.1 PASOA Architecture . . . 17

1.3.2 Kepler Project . . . 18

1.3.3 The VIEW System . . . 19

1.3.4 Matrioshka . . . 21

1.4 Applications . . . 21

1.4.1 Trust Assessment . . . 22

1.4.2 Provenance Re-execution and Validation . . . 23

1.4.3 Workflow Reduction . . . 24

1.4.4 Data Replication . . . 25

1.4.5 Informational Use of Provenance . . . 25

1.5 Problem Description . . . 25

1.5.1 Challenges . . . 27

1.6 Contributions and Thesis Overview . . . 29

2 A Multi-Functional Architecture Addressing Workflow and Service Challenges Using Provenance Data 34

2.1 Introduction . . . 35

2.2 Motivation and Requirements . . . 38

2.3 Multi-Functional Provenance Architecture 42
2.4 Related Works 46
2.5 Methodology 48
2.6 Evaluation Design 50
2.7 Conclusion 51

3 Evaluating Workflow Trust using Hidden Markov Modeling and Provenance Data 53
3.1 Introduction 54
3.2 Related Works 55
3.3 Hidden Markov Modeling for the Evaluation of Workflow Trust 57
3.4 Methodology 59
3.4.1 Trust Model Assessment 63
3.4.2 Cases with Dynamic or Parallel Sections 66
3.5 Implementation 67
3.6 Case Study 70
3.6.1 Investigation of the Stationary Assumption 75
3.7 Conclusion 79

4 Extracting Workflow Structures through Bayesian Learning and Provenance Data 81
4.1 Introduction 82
4.2 Related Works 84
4.3 Methodology 87
4.3.1 Bayesian Structure Learning Methods 87
4.3.2 Modeling the Workflow Model Extraction Problem as Bayesian Structure Learning 89
4.3.3 Parents and Children (PC) Algorithm 92
4.3.4 Max-Min Parents and Children (MMPC) Algorithm 95
4.4 Implementation 97
4.5 Case Study and Performance Evaluation 98
4.5.1 Three Scenarios 98
4.5.2 Experimental Results 103
4.6 Conclusions and Future Work 103

5 Automatic Service Composition Using POMDP and Provenance Data 106
5.1 Introduction 107
5.2 Related Work 110
5.3 Modeling and Methodology 113
5.3.1 POMDP Dynamics 113
5.3.2 Web Service Composition as POMDP 115
5.3.3 POMDP Algorithms 120
5.4 Implementation and Case Study 121
5.4.1 Implementation 121
5.4.2 Case Study 122

5.5 Experiments and Results 124
5.5.1 Verification of Method 124
5.5.2 Scalability Analysis 125
5.6 Conclusion 128

6 Conclusions and Future Directions 130
6.1 Conclusions 130
6.2 Future Directions 131

LIST OF TABLES

3.1 Trust level decision table, L, M, and H denote Low, Medium, and High. This table represents how, in our implementation, the combination of several QoS values is mapped to a trust state. 68
3.2 Range Level of the QoS parameters Availability 69
3.3 The average of the values of the QoS parameters generated for the scenario. 72

4.1 Comparison of some workflow mining approaches [Tiwari et al., 2008]. 86

LIST OF FIGURES

1.1 Service interaction in a service-oriented environment [Srinivasan and Treadwell, 2005] 2

2.1 Multi-Functional Provenance Architecture. 43
2.2 HMM for Service Selection. 50

3.1 Basic HMM. 58
3.2 A sample workflow and the HMM for workflow trust evaluation. 61
3.3 A sample workflow scenario - KDD Process. 71
3.4 Transition matrix of the example 72
3.5 The state transition diagram showing the transition probabilities for the above example learnt. 73
3.6 The change of the trust state probabilities over time using an HMM. 74
3.7 Comparing the average trust level of HMM vs. NSHMM for 5 to 25 numbers of services with increments of 5. 78

4.1 Implementation Model. 97
4.2 Case Study 1: a sequential workflow scenario. 100
4.3 Case Study 2: a workflow with parallel parts. 101
4.4 Case Study 3: a more complicated scenario with a split-or, a split-and and a join. . 102
4.5 Case Study 3: Workflow Results of Modified PC and MMPC 102
4.6 Case Study 3: Workflow Results of PC and MMPC 102
4.7 Performance Comparison of MMPC and modified MMPC 104
4.8 Performance Comparison of PC and modified PC 104

5.1 Case study scenario 123
5.2 Policy graph generated by the POMDP solver 125
5.3 Performance evaluation for different numbers of concrete services with a constant number of abstract services 126
5.4 Performance evaluation for different numbers of abstract services with a constant number of concrete services 127
5.5 Performance evaluation with regard to various POMDP algorithms 127

List of Abbreviations

DAG	Directed Acyclic Graph
EM	Expectation Maximization
GSP	Generalized Sequential Pattern
GIS	Geographic Information System
HMM	Hidden Markov Model
IBRG	Image Bioinformatics Research Group
IP	Internet Provider
LP	Linear Programming
MCPP	Multi-Constraint Path Problems
MDP	Markov decision processes
ML	Maximum Likelihood
MMPC	Max-Min Parents and Children
MOP	Multi-Objective Programming
OWL	Web Ontology Language
PASOA	Provenance-Aware Service Oriented Architecture
PC	Parents and Child
POMDP	Partially Observable Markov Decision Process
QoS	Quality of Service
RDF	Resource Description Framework
SAW	Simple Additive Weighting
SOA	Service Oriented Architecture
SWS	Semantic Web Services
VIEW	VIsual sciEntific Workflow
WSDL	Web Service Description Language
XML	eXtensable Markup Language

CHAPTER 1

INTRODUCTION

In this chapter, the basic concepts regarding the area of study are briefly introduced. These concepts include service-oriented architecture, workflow, and provenance. First, information on service-oriented environments are presented, and it is explained how services can be orchestrated by exploiting workflows in service-oriented systems. Later, the requirements for considering the origins and routes of data and its impact on service oriented systems is discussed. Next, the concept of provenance is explained along with its application areas. The architecture of different provenance systems, provenance lifecycle, and applications of provenance data are the topics which will be investigated further in this chapter. Finally, we discuss the challenges in service oriented environments and present our methodology in tackling these problems using provenance information and provide an overview of the research that was conducted in this thesis.

1.1 Introduction

1.1.1 Service Oriented Architecture

Service Oriented Architecture (SOA) [Newcomer and Lomow, 2005] is an architectural discipline appropriate for the infrastructures in which consumers or providers need to interact via services across distributed domains of technology and ownership. Services are the building blocks of such infrastructures. The fact of service-orientation aims at the loose coupling of services with operating systems, programming languages and other technologies that underlie the applications being run on such environments. Services play either the role of a provider, which publishes its interface and access information to service registries, or the role of a consumer (or requester), which locates service providers and binds to a provider to invoke its operations [Srinivasan and Treadwell, 2005]. Services can be either individually useful as atomic services, or might be composed to provide higher level composite services and other functionalities. They communicate with their clients by exchanging messages and advertise their properties, such as their capabilities, policies, interfaces and communication protocols.

A simple service interaction cycle is shown in Figure 1.1. A service provider advertises itself through a registry service. A registry includes a list of services and the functionality they provide. A client, i.e., service requester, queries the registry to search for a service that satisfies its requirements. The registry returns a list of services matched with the request and the requester selects one.

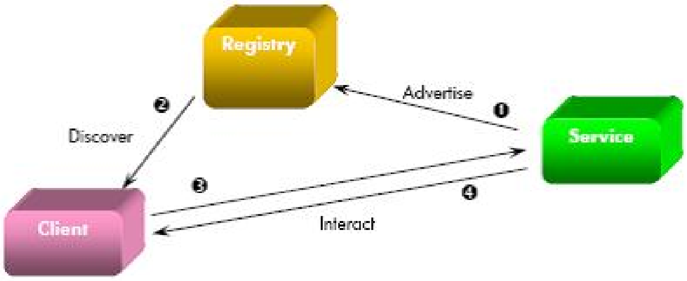


Figure 1.1: Service interaction in a service-oriented environment [Srinivasan and Treadwell, 2005]

1.1.2 Workflow

A Workflow can be defined as a sequence of tasks which are put together in a special order to achieve a goal. The followings are different examples of a workflow:

- In machine shops, particularly job shops and flow shops, the flow of a part through the various processing stations is a work flow. [Wik, 2009]
- The procedure of ordering a product from receiving the order to its shipment is an instance of a workflow process.

Workflows can be presented in different levels of complexity. At the most basic level, the tasks can be specified in a linear order. Each task transforms a data or control object to the next task. At the next level of complexity, a workflow can be described by an acyclic graph where the nodes represent a task and the edges represent the dependencies between tasks. Also, a workflow can be represented as a cyclic graph, where the cycles represent the iteration control mechanism.

Workflow systems can be divided into two groups, namely scientific workflow and business workflow systems [Tan and Zhou, 2013]. Scientific workflow systems adopt a dataflow model and the order of executions in these systems is the same as the order of the flow of data through the workflow. Business workflow systems, on the other hand, specify complex control flows, the orders in which workflow tasks are executed.

In case of the service-oriented environment, the workflow is defined as the automation of the processes and involves the orchestration of a set of services, agents and actors that must be combined together to solve a problem or define a new service. The workflow graph describes a network where the nodes are services and the edges represent messages or data streams that channel work or information between services. Each node processes a stream of messages and pushes the result streams into its connected neighbors [Fox and Gannon, 2006].

1.1.3 SOA and Provenance

A service-oriented infrastructure exploits and shares resources for the purpose of problem solving. This has led to a growing demand for tracking, recording and managing data sources and their derivation process.

In such environments, great numbers of workflows are executed to perform computational and business tasks. The workflow activities are run repeatedly by one or more users and large numbers of result data sets in the form of data files and data parameters are produced. As the number of such datasets increases, it becomes difficult to identify and keep track of them. In addition, in large scale scientific computations, how a result dataset is derived is of great importance as it specifies the amount of reliability that can be attributed to the results. Thus, information on data collection, data usage and computational outcome of these workflows provide a rich source of information [Altintas, 2008]. Capturing this information, which is regarded to as provenance information, is a significant advantage of using workflows. To put it in a nutshell, provenance is the metadata that tracks the steps by which the data was derived. It can provide significant value in data intensive scenarios. It facilitates determining data dependencies, following the steps in the workflow design, validation of the workflow results, workflow re-executions, and error recovery. Provenance also enables users to trace how a particular result has been arrived at by identifying the individual services and aggregation of services that produces such a particular output.

Many Grid-based applications for provenance exist that have different requirements but essential needs for provenance. For example, in aerospace engineering, components are combined for the purposes of simulation, pre- or post-processing and visualization. Provenance is then required to maintain a historical record of the output of the components so that the customers can use the end results of the simulations. In addition, provenance data of the aircraft's structuring phases is kept for a long time period for the purpose of selling them to other countries [Moreau and et al., 2008]. Another example of provenance concerns organ transplant management. Medical information systems rely on large amount of data. Organ transplant processes are an example of these information systems which benefit from Grid technologies because of the large number of patient records and tissue banks. In these scenarios, provenance data can be used to trace back previous decisions to identify whether the best organ match was made or to aggregate partial results from searches in different centres and maintain the validity of the results [Moreau and et al., 2008].

The concept of provenance is a broad concept and is opening its way throughout different areas of computer science including databases systems [Greenwood and et al., 2003], bioinformatics [Greenwood and et al., 2003], sensor networks [Ledlie and et al., 2006], social networks [Golbeck, 2006a], etc. In the following section, the concept of provenance, its origin, its different areas of application along with provenance architectures and applications are discussed.

1.2 Provenance: Definition, Systems, and Applications

In this section, we focus on the concept of provenance. First, the definition of the word provenance in the dictionary is presented. Later, the history of this word and its applications in different sciences as well as computer science are explained. Next, the general architecture of provenance systems is presented and information on current provenance systems, their life cycle and components are provided. And finally, the various applications of provenance data are studied.

1.2.1 Provenance: Definition and Examples

The word provenance comes from a French verb “provenir” which means “originate”. In the Oxford English dictionary this word has been defined in two different ways [Simmhan et al., 2005]:

- The fact of coming from a particular source; origin.
- The history of ownership of a valued object or work of art or literature.

Two different understandings of “provenance” can be achieved from the definitions above. “Provenance” can refer to either the source (or derivation) of an object or a record of such a derivation.

In real world problems, some organizations, like cultural and heritage ones, record the origin of certain classes of objects. It is particularly important in museums where it can be used to help understand the origin, transmission and the chain of ownership of the objects. Provenance data can denote the location and date on which a particular object was found. This information is very important for archaeological and geological purposes, where this data can be combined and re-analyzed to help in developing deeper understandings. For example, provenance research in archeology can help in discovering where raw materials are mined or manufactured. Some methods like neutron activation analysis (NAA) are used to trace objects back to their first place of origin. The same principles are applied in geology to find the composition of sandstones, etc. Provenance has also become an important topic in art history, where it can help in investigating the chain of ownership for providing information on the history of individuals. Provenance has also found its way through the archival science. Archival records are the consequences of activities that are defined by organizational functions. They started the archiving of record descriptions and later provenance authority records. [Day, 2005]

It can be seen that the concept of provenance is based on the principle of archiving. In scientific experiments, provenance information can help in understanding the results of experiments, which is done by investigating and following the steps, data and reasoning that has led to that result. Traditionally, publications were used to maintain and represent this information, but with the increase in the complexity of analysis and the amount of data, it is desirable to capture provenance data automatically and systematically.

Scientific domains exploit various forms of provenance in different domains and for different purposes. For example, Geographic Information System (GIS) standards require a description of the lineage of the data products to help the users in deciding whether the resulting dataset meets their requirements. [Simmhan et al., 2005]

In materials engineering, it is essential to have the pedigree of the materials used for designing critical components for the purpose of auditing and preventing system failures [Simmhan et al., 2005]. In life sciences research, sharing of the biological and biomedical data and their transformation record provides information about the credit of the authors and the context in which this data can be used. Uniform astronomical provenance data is shared publicly through archives to help astronomers estimate the trust that can be placed on them.

Other than the scientific domain, provenance data has applications in the business domain. A great amount of businesses deal with bad quality data. All data, including bad quality ones, are gathered together in data warehouses. Later, business analytic and intelligence tools are used to mine these data and help in decision making. In such an environment, lineage information is used to trace the data in the warehouse and discover the source and origin of data and find additional characteristics of data sources which are not available in the warehouse. It is also used to trace faulty data back to the source of the error and to make corrections to it; therefore, in computer science, provenance is mainly concerned with data [Simmhan et al., 2005].

Provenance data is used in different architectures and for different purposes. SOA is an area in which provenance is widely used. Grid and web services provide a rich platform for the scientific community. The transformation of data in such environments is usually specified in the context of workflows where different services of the workflow represent the transformation processes which receive the data as input and produce a transformed version of that data as output [Simmhan et al., 2005]. The execution details of the workflow are traced automatically and the metadata of the input and output data of each service is gathered. Later, users can provide and add additional metadata on the data or the execution process. In the context of SOA, "Provenance" was given the following definitions:

- "Provenance of a piece of data is the process that led to that piece of data." [Grawth et al., 2006]
- "Provenance provides explanations about who, how, and what resources were used in a process, and the processing steps that occurred to produce a result." [Rajbhandari et al., 2008]
- "Provenance is described as the documentation of a process that led to a particular result."

Provenance in SOA is mostly process-oriented, as the data lineage is deduced indirectly and the deriving processes are the primary entities for which the provenance is collected. The input and output products of these processes determine the data provenance. Workflow provenance is also called coarse-grained provenance, as it records a complete history of the derivation of some data and involves not only tracking of the interaction of programs but also the information of the external devices, such as sensors, cameras or other data collecting equipments. It may also involve the recording of human interactions with processes.

Database architecture is another architecture in which provenance has substantial applications. Recently, provenance has been shown to be important to understand the transport of annotation in database views, data integration, view update and maintenance, and probabilistic databases. A data product, which includes a table, a view, or a tuple, can have a lineage that can be traced back through a series of functions and queries. Other than this, dataflow graphs can be built and executed by the database as part of the query, which is analogous to the workflows in service oriented architectures. Again, annotations can be added by users about data sources and queries. The techniques that are usually used to trace the lineage are query inversion and function inversion [Cui and Widom, 2000]. In the context of databases, provenance is determined by the following definitions:

- "Provenance describes the source and derivation of data." [Buneman and Tan, 2007]
- "Provenance describes how a data item came to existence, particularly if it was derived using other data at some point in time." [Widom, 2006]

In contrast to the process-oriented model, provenance in database architecture is explicit and data-oriented. In this model, lineage metadata is specifically gathered about a data product. A database's provenance is also referred to as fine-grained provenance, as it concerns derivation of part of the resulting data set. A fine-grained provenance is an account of the derivation of part of the resulting data set.

Provenance of computational tasks can be in two different forms, including prospective provenance, which captures the specifications of the individual tasks, and retrospective provenance that captures the process and the steps executed as well as the information about the environment.

Provenance can also be viewed from the two perspectives of execution or service provenance. Execution Provenance relates to data recorded by a workflow engine during a workflow execution. It identifies what data is passed between services, what services are available, and how results are eventually generated for particular sets of input values, etc. Using execution provenance, a scientist can trace the "process" that led to the aggregation of services producing a particular output. Service Provenance relates to data associated with a particular service, recorded by the service itself or its provider. Such data may relate to the accuracy of results a service had produced, the number of times a given service has been invoked, or the types of other services that have made use of it. A service provider may make such information available to other users to enable them to select services that are more likely to produce the output they desire. [Moreau et al., 2013]

1.2.2 Provenance Application Areas

It was mentioned that provenance has found its application throughout different areas of science. In this section, the main areas in which provenance is widely used and the various provenance research that was done (or are being done) in these areas are introduced. Provenance is given different definitions according to the domain it is applied to. Therefore, in the following, the definition of provenance for different application areas is also discussed.

Databases

In databases, provenance is often used to recover the source data from the output data, avoid duplication of data, and assess the quality of databases. [Moreau et al., 2013] presents some applications of provenance in databases. It provides information about the inspiration behind the current provenance research works done in the area of databases. In [Wang and Madnick, 1990], a model was proposed in which provenance in the form of annotations was carried along with source attributions in the result of the queries. [Woodruff and Stonebraker, 1997] proposes the idea of merging database management systems with the capability of returning fine-grained provenance, which can be done by allowing weak function inversions. Applying weak inversions to functions returns some approximation of provenance data as the results of the function. Some research, [Cui, 2000], investigated provenance computation by analyzing relational algebra and its extensions. The Trio project [Greenwood and et al., 2003] introduced a new database system that provides management as well as querying facilities for the lineage of data as well as the data itself. One of the recent applications of provenance in this project is toward probabilistic databases [Fuhr and Rolleke, 1997], which are databases with uncertainty. In such databases, provenance is used to determine whether the sources of tuples are independent. The provenance of tuples can also help in capturing the set of possible instances in the result of a probabilistic query. Provenance can also help in describing trust policies in collaborative data sharing systems. For this purpose, provenance is used to infer the relationships between the source and target data in data exchange or integration scenarios. The value of a curated database relies on their provenance. Data is entered and copied manually from other sources; therefore, provenance data determines the reliability and the trust that can be put on data. As mentioned before, annotations are one kind of provenance which add to or mark up existing data. Capturing annotations of database data and propagating them from source to output is desirable for many databases. Current emerging research in the area of databases concern the application of provenance in the analysis of update query languages [P. Buneman and Vansummeren, 2008].

Bioinformatics

e-Science is the use of electronic resources such as instruments, sensors, databases, computational methods, and computers by scientists working and collaborating in large, distributed teams to solve scientific problems. Data derived from e-science experiments is not valuable for further use if the origin and provenance of that data is not known. For example, an *in silico* experiment, which exploits computer-based information repositories and analysis techniques to test a hypothesis, derive a summary, or search for patterns, could use an open source workflow enactment engine. The myGrid [Greenwood and et al., 2003] project enables generation and management of provenance data. It is developing service-based middle-ware to support construction, management, and sharing of data-intensive experiments in biology. The project keeps track of the derivation path provenance. These include the details of the execution process; information about the input data; information about the workflow description, as well as all the actions of the user.

The derivation provenance can be exploited for:

- Repeating and validating an experiment; this requires the provenance to be expressive enough.
- Learning from the provenance history and disseminating best practices.
- Helping the scientists to know if the experiment they wish to run or their hypothesis has been tested before.
- Automatically re-running an experiment when there is a change in data, tools, or data repository.

In many scientific databases, specifically the bioinformatics ones, there are database curators who manually select, organize, classify and annotate data. The value of these curated databases relies on its provenance.

The numbers of available biological databases are rapidly increasing; therefore, obtaining knowledge about a gene or protein from this data, requires going through information gathering processes and navigating between databases. In order to identify how these data resources are linked with each other, the Image Bioinformatics Research Group (IBRG) project [Zhao, 2009] proposes recording the provenance of datalinks to maintain and link between related data items, which helps in bringing trust to the data web by providing evidences for links or tracing how the data links have been updated and maintained.

Sensor networks

In sensor networks, the events that are read from the sensors might be consumed immediately or stored for later reasoning and analysis. In many of the situations, data from distinct sensor networks are collected and combined for better reasoning. The collected sensor data is often useful for historical analysis long after its collection. Following the example provided in [Ledlie and et al., 2006], traffic data received from the London Congestion Zone's sensors can be used immediately to ticket non-paying drivers. It can also be combined geographically with data from other cities to gather a broader picture of traffic. In addition, this historical traffic data can be merged with historical weather data to gain deeper insights.

In such environments, naming and searching for sensor datasets is necessary. [Ledlie and et al., 2006] investigates the data requirements such as storage, naming and indexing for sensor networks and argues how these requirements can be targeted using provenance data. The descriptions of sensors and annotations of data that are stored in sensor networks include information regarding the replacement of sensors, software upgrades of sensor devices, etc. For the purpose of annotations, the right attributes to index this type of data should be selected. In addition, the granularity at which indexing should be done plays an important role. Provenance can be helpful in indexing the names given to sensor readings by presenting them with identifying information which provide a unique identifier for that data. The specific details of the provenance metadata representation are application-specific. As the complete provenance of reading sensor data is large, querying such data will be more than a simple looking up a name, and requires searching for datasets based on subsets of the attributes and values found in provenance metadata. Therefore, indexing structures in sensor data storage systems should provide for efficient search and efficient recursive and transitive queries.

Social Networks

Social networks are a popular phenomenon on the web and the semantic web is rich with social network information. In social networks, there exist nodes, which are individuals or organizations, and the relationships (friendship, kinship, common interest, etc.) between the nodes make up the ties. Users in such an environment can use other ontologies to get more information about their social connections, such as the type of relationships or the trust value of other nodes. Ontologies are formal representation of knowledge by a set of concepts and the relationships between those concepts. Ontology languages are formal languages which are used to represent ontologies. The most common ontology languages in semantic web include Resource Description Framework (RDF) [RDF, 2004] and Web Ontology Language (OWL) [OWL, 2004]. OWL is based on RDF, which is designed as a metadata data model and is based on XML [XML, 2013]. Trust and provenance can be integrated in such networks and the trust relationships can be inferred using trust annotations and provenance data. Trust annotations about nodes, i.e., services, are made by users and stored in a provenance store. These annotations include social relationships of services and are represented using a vocabulary that is described by ontologies. The annotations are then used to infer how much two users trust each other. For this purpose, the paths that are connecting these users are investigated and the values of the trust of nodes along these paths are gathered [Golbeck, 2006b]. For inferring trust, different projects are working on new algorithms to find shorter paths, which lead to more accurate information, and also puts a limit on the path sizes. The computation of trust values in social networks is a use of provenance and annotations together, and the resulting trust values are applicable for personalizing contents. Having provenance information for the annotations which are found on the semantic web and a social network with trust values, the information that is presented to the user can be ranked, sorted or aggregated according to trust. Examples of such scenarios include the FilmTrust [Golbeck, 2006a], which is a website with a social network in which users can rate movies and write reviews on films. The data in this system is all stored with semantic web annotations and users rate the trustworthiness of their friends. These trust values are used to present personalized views of movie pages.

1.3 Provenance Systems

In this section, the architecture of provenance systems along with their lifecycle are investigated. The main focus is on current provenance systems and the way they are designed and implemented. The on-going and previous research on different steps of the provenance lifecycle are studied and the advantages and disadvantages of each approach are discussed. Generally, a provenance management system is composed of three components which are described as:

1. Capturing mechanism,

2. Representation model,

3. And an infrastructure for storage, access and queries.

The capturing mechanism is responsible for gathering provenance information and needs access to relevant details of computational tasks, such as the steps they follow, execution information and user-specified annotations. The representation model provides support for representing retrospective and prospective provenance as well as annotations which as mentioned earlier are considered part of provenance information. The model should present information on process and data dependencies. Despite the base commonality in their functionality, provenance models vary according to domain and user needs [Freire et al., 2008]. It is very advantageous if the model is structured with a set of layers to enable configurable representations. The layered model also leads to simpler queries and better results [et al., 2007].

In the following parts of this section, some of the most important provenance architectures and their components are discussed. The provenance systems to be described include the Provenance-Aware Service Oriented Architecture (PASOA) project [Groth et al., 2005], which aims at investigating the concept of provenance and its application for reasoning about data and services in the context of science. The architecture presented by this project designs a distributed cooperation protocol for generating provenance data in workflow enactment. The architecture which is discussed afterwards is mostly based on the provenance system of the Kepler project [Kep, 2013]. The Kepler project is dedicated for furthering and supplying the capabilities and awareness of the free and open-source scientific workflow application. The provenance architecture of this system focuses more on data provenance, and investigates the components required for a provenance system with regard to data collection, data usage, and data usage feedback. In the third architecture, the provenance manager of the VIsual sciEntific Workflow (VIEW) system is presented. VIEW is a workflow system which uses semantic web technology to represent, store and query provenance metadata, leading to an interoperable and extensible provenance system. It also supports the visualization of provenance graphs.

Finally, the last architecture presents a Semantic Web Services (SWS) based system architecture for modeling, capturing and querying augmented provenance in SOA. It uses ontologies for provenance modeling and SWSs for capturing execution-independent metadata.

1.3.1 PASOA Architecture

PASOA is a project which aims at investigating the mechanisms necessary to support the notion of provenance in Grid and web service environments. According to PASOA, the provenance lifecycle is composed of four different phases which include:

1. Provenance creation.
2. Provenance recording.
3. Provenance querying.
4. Provenance management.

In the provenance creation phase, the provenance data, the concepts from which a result has been achieved, are created. The generated provenance data is then collected and stored in a provenance store, which is usually one or a combination of databases. The way this data is represented depends on the way it is stored and vice versa. In some systems, it is represented as texts and stored in files, while in others it might be stored in databases.

The recording phase of the provenance lifecycle results in a set of provenance data represented in a determined model in the provenance store. This data constitutes a documentation of workflow executions and provides information from which a representation of the provenance data, in which the users are interested in, can be derived. The gathered provenance can then be queried. In this phase, the provenance queries select, scope, and filter out a subset of provenance data and make them available in some representation. In order to support complex querying functionality, the provenance data should provide complete and detailed enough information [Grawth et al., 2006].

According to the above lifecycle, a provenance system can be defined as a computer system that deals with all issues of recording, maintaining, visualizing, reasoning, and analysis of the documentation of the process that underpins the notion of provenance. Such a system will be based on a provenance architecture that specifies the different roles of the system, their interactions, and their provenance representation [Groth et al., 2005]. The actors involved in the lifecycle of such a system define different roles. They are categorized as application actor, provenance store, recording actor, querying, and managing actors. These actors undertake the responsibilities for executing the application's business logic, managing and providing access to the recorded provenance data, submitting data to the provenance store, and issuing provenance queries to the provenance store.

In order to facilitate the recording and querying of provenance, PASOA has developed PReServ [PRe, 2005] an implementation of the Provenance Recording Protocol.

1.3.2 Kepler Project

The research presented in [Altintas, 2008] investigates the lifecycle of scientific provenance systems from the data provenance perspective. The procedure starts with data collection of the workflow design and execution steps. This data include information about changes to workflow, workflow versions, execution time parameters, inputs, outputs, intermediate results, etc. As the provenance users fall into different categories and each have different requirements, provenance recorders need to allow for a customized data collection through interfaces. The architecture emphasizes a three stage recording model which starts during the design phase of a workflow, and continues during the experiment operation. The recording process still continues even after the workflow results and provenance information are published. The final stage enables verifying the scientific impacts the workflow has made.

The recorded provenance data is then exploited for different purposes, such as monitoring workflows, re-running them, comparing different versions of them, fault-detection, etc. The data can also be analysed and queried to find associations between workflow inputs and outputs, comparing the results and performances of different workflow models, etc. The architecture also supports a data usage feedback which learns what has worked for different runs. The feedback information is used for similar design efforts in future for recording purposes.

The Kepler workflow system provides a reporting suite [Kep, 2011] which includes the ability to create reports displaying workflow results, capture provenance of workflow execution, and manage workflow runs.

1.3.3 The VIEW System

The work in [Chebotko et al., 2007] discusses a new workflow management system that supports provenance. It introduces seven architectural requirements that such systems should have. These requirements include support for user interaction, reproducibility, heterogeneous service and tool integration, heterogeneous data product management, high-end computing support, monitoring and interoperability. The proposed architecture is composed of four main layers:

- Operational layer: with subcomponents of task, provenance and data product management;
- Task management layer: which supports efficient management of tasks, data products, and provenance metadata;
- Workflow management layer: with subcomponents for monitoring workflows and workflow engines;
- Presentation layer: considered for visualization and design of workflows.

In an overall view, the workflow engine implemented in this project, supports the execution of a workflow and the collection of provenance metadata in semantic languages. This metadata is stored in the provenance server which provides 3 main functionalities: setup, recording and querying. The setup creates a relational database in MySQL and generates its schema. The recorder uses an interface engine to infer new data triples based on predefined semantics and additional interface rules and stores them in the database. The Query component provides query interfaces to access the provenance stored in a database. [Chebotko et al., 2007]

The architecture also supports a provenance manager which is composed of three layers: provenance model, relational model, and model mapping. The Provenance model layer represents execution provenance domain ontologies. Web Ontology Language, OWL [OWL, 2004], is used for expressing these ontologies and the SPARQL [SPA, 2004] Protocol and RDF Query Language, for describing queries. The relational model layer includes relational provenance storage and SQL is used for querying purposes. The model mapping layer, which is a layer between the two other layers, can either map ontologies to database schemas or map provenance metadata into relational tuples and store them in the database. This layer is also able to map SPARQL queries into relational queries in SQL.

1.3.4 Matrioshka

An architecture, Matrioshka, for controlling provenance in distributed systems is presented in [Cruz et al., 2008]. This architecture, which has been implemented, consists of a set of services that can be coupled to workflow management systems. Provenance is described in terms of data flow graphs with nodes representing computations and edges representing data dependencies. Heterogeneity, scalability, querying and different granularity levels are claimed to be supported in the architecture. The whole architecture is composed of a provenance broker which is responsible for caching, security and brokering, a provenance browser, provenance eavesdrop, and provenance repositories. The provenance broker gathers data through event notifications. It also provides data transformations and routes of the gathered data. The browser is a web interface that combines data into a tool for presenting and searching provenance. The provenance eavesdrop service generates event notifications that publish details about the data being stored, the execution status of the remote application, the location of the output results, execution times, security warnings, etc. A uniform data query is offered by the broker. The provenance broker and eavesdrop are designed to be plugged into workflow management systems and compliment them so that they do not need to be changed.

1.4 Applications

In the previous sections, we provided information regarding the concept of provenance, different scientific areas which apply provenance, the architecture of provenance systems and the different phases these systems are involved in. Producing provenance data is of no use if it is not being exploited. In this section, the applications of provenance – along with the various types of reasoning and inference that can be applied to it – are presented.

1.4.1 Trust Assessment

Provenance data can be exploited for evaluating trust of workflows. Sequences of tasks, each represented as a service, are put together in a special order to make a workflow for the purpose of solving a problem or defining a new service. A trust value is associated to each service that signifies the Quality of Service (QoS) provided by the service provider for that service. The QoS is distinguished by several parameters, such as response time, availability, reliability, status, etc. The trust degree of a service is the general estimation of the QoS values of that service and plays an important role in service consumer selection. Providing the trust value for each service, the overall trust value of a sequence of services can be evaluated.

In [Rajbhandari et al., 2006], a trust architecture is presented which exploits provenance data for assessing service trust. The architecture utilizes both process and actor provenance for trust evaluations. Three types of trust are distinguished, and a decision tree is modeled based on these three categories to compute the value for the trust of the workflow.

Extending the work that was done in [Rajbhandari et al., 2006], a fuzzy model for calculating the trust value for a workflow is presented in [Prat and Mandick, 2008]. In the decision tree model the result value for trust was binary, while in this work a degree of trust was considered. The authors have added an analysis tool to the model which is a Jess rule engine [JES, 2008]. For each analysis node of the decision tree, the results are sent to this tool and are mapped to fuzzy membership functions. The final decisions are made using Jess inference rules.

In another work, presented in [Rajbhandari et al., 2008], it is discussed how data believability can be inferred from provenance information. Data believability, an important aspect of data quality, is defined as the extent to which data is accepted or regarded as true, real and credible. As can be inferred from this definition, believability of a data value depends on its origin and subsequent processing history which is provided by provenance information.

Believability of data is a composed concept of the trustworthiness of source, reasonableness of data, as well as temporality of data. To compute all the three sub-dimensions, [Prat and Mandick, 2008] provides a provenance model consisting of a database schema for processes, data values, valid time of facts, transaction times and trustworthiness of agents. The quality of any data value depends on the quality of source data and the processes. In order to assess believability, first the believability of data sources is computed using the provenance information. Secondly, the believability of process results are evaluated by measuring the weight of each data value and finally the global believability of dimensions are assessed by averaging the temporal believability of all values in the provenance for that data item.

1.4.2 Provenance Re-execution and Validation

In many different e-Science areas, experiments involve many distributed services maintained by different organizations. After finishing the experiment, it is important to verify that the experiment was performed correctly. There is no existing standard framework for validating experiments in today's e-Science frameworks. Two commonly used forms of validation include static and dynamic validation [Miles et al., 2005]. Static validation operates on workflow source code, while dynamic validation is performed at run-time. However, it is sometimes necessary to validate an experiment after it has been executed. A provenance-based approach for workflow validation facilitates this. In order to be able to validate and re-execute the workflow, sufficient provenance data should be recorded to be able to re-create any dataset transformation. In [Szomszor and Moreau, 2003], a provenance system is presented which consists of validation and browsing of provenance data. Validating the data is possible by doing reasoning on provenance data and it is used to verify, for example, whether the services still produce the same results and the workflow is still valid. For this purpose, the workflow is re-executed iteratively using the inputs that were recorded in the provenance data. The outputs that are produced by the re-invocation of services are compared with the output stored in the provenance trace. Provenance validation facility verifies whether the results produced by a previous execution of a workflow are still up to date.

1.4.3 Workflow Reduction

Many scientific applications involve large amounts of computations operating on large volumes of data. As the number and the size of computational jobs increase, the management of data becomes more difficult without considering automatic refinement operations. One of the possible refinement operations is called reduction. A reduction is performed when unnecessary tasks in a workflow are pruned out. This is possible when the datasets to be generated by these tasks have already been computed previously and it is more efficient to access them than to re-compute them. The system provided in [et al., 2007] is also capable of reducing workflows before the execution by exploiting the recorded provenance data. The workflow can be reduced by searching the provenance data for the tasks that have been run previously and whose results are available. The system then registers the jobs for which no result is available for their execution for the purpose of future exploitation.

1.4.4 Data Replication

As provenance information includes the steps followed to derive the datasets, it can be used for recreating the datasets. Having sufficient information regarding the operations, data sources and parameters, the same data can be generated automatically using the provenance data. It is important to consider that the availability of similar resources is a requirement for getting the same results. Thus, it is possible and sometimes cost effective to use provenance data as a means of replicating the data instead of transporting or storing it. In order to recreate the process, it is required to access the same data and process, as well as the processing environment.

1.4.5 Informational Use of Provenance

Dataset discovery, knowledge extraction, and metadata exploration are generic applications of provenance. Searching for the source of data or processing the steps used to produce the data using querying mechanisms are the common uses of provenance data.

1.5 Problem Description

In this chapter, the concept of provenance was defined and its application in different areas of computer science including databases, social networks, sensor networks, and Bioinformatics was outlined. The research provided in this chapter shows that provenance is a broad concept and various areas of computer science can take advantage of the benefits added by providing support for it.

From the provenance applications discussed in this chapter, it is observed that in comparison to the efforts made to gather and store provenance data, not much research has been done on discovering useful applications for the collected information. Therefore, it seems necessary to explore and discover generic applications of provenance. As long as it is known what provenance types are stored and gathered in provenance systems, it is possible to find applications that are common for all types of provenance data, no matter to what specific area that data belongs.

Learning is one of the unexplored applications of provenance. A large store of the previous executions of services and workflows, as well as their specifications, provides an appropriate data set for learning and knowledge discovery. The provenance data can be explored using data mining and pattern recognition methods to discover the patterns of interest in data. The store is also a suitable source for learning probabilities. Therefore, probabilistic learning methods can be used to produce the required parameters for the probabilistic decision making processes in order to learn the workflow structures or compose workflows. To assess the probability values for these processes, the Maximum Likelihood (ML) [ML, 2013] method or similar approaches can be applied on the provenance data. ML learning is a data analysis approach for determining the parameters that maximize the probability (likelihood) of the sample data. It is important to mention that this is based on the assumption that the provenance data on which reasoning is performed does not include missing data.

To be able to find the probabilities in case of missing data, the Expectation Maximization (EM) [Boreman, 2009] learning algorithm can be used. The EM algorithm is an efficient iterative procedure to compute the ML estimate in the presence of missing or hidden data. Using this algorithm, the missing values are first predicted based on assumed values for the parameters. Later, these predictions are used to update the parameter estimates. The sequence of parameters converges to ML estimates and EM implicitly averages over the distribution of the missing values.

1.5.1 Challenges

Applying learning and knowledge discovery methods to provenance data can provide rich and useful information to workflows and services. Therefore, in this thesis, the challenges with workflows and services are studied to discover the possibilities and benefits of providing solutions by exploiting provenance data. These challenges mainly include composing services and creating workflows automatically, assessing the performance of workflows and services, discovering workflow models, and repairing them. In the following, we discuss these challenges:

1. Workflow Assessment and Evaluation

   Most research on workflow systems focus on prediction, tracking and monitoring of workflows, and not on evaluation of these processes. The few works which studied evaluation, accomplished a very narrow research goal aiming to improve performance or fault tolerance of workflow systems [Aiello, 2004]. As the provenance information maintains the records of previous execution details of workflows, it provides the facility to analyze, assess, and evaluate the behavior of a workflow as well as its performance in terms of trust, usability, and QoS assessment. The performance of a workflow, its believability, improvements, and its future trend, etc. can be analyzed and evaluated through provenance data.

2. Mining Workflow Structures through multiple perspective

   Workflow mining discusses techniques for acquiring a workflow model from a workflow log. Workflows can be investigated from many perspectives: functional, behavioral, informational, organizational and operational. In case of the behavioral perspective, which looks at control flow, workflow mining is done by following the order in which events for tasks are stored; for the informational perspective which looks for data flow, usually inputs/outputs are being used; in case of the organizational perspective, participants of tasks and their roles are being discovered in workflow mining. The workflow mining methods currently use the event-logs for discovering the patterns and mining the workflows. Event logs keep track of small amount of information which is not enough for mining workflows with regard to all the mentioned workflow perspectives. Instead, the data presented in workflow provenance provides a much stronger reasoning and mining ground.

3. Service Composition and Selection

Composing and selecting services dynamically for the purpose of achieving a goal is a problem of interest in service-oriented environments, as a composition of services can provide higher functionalities compared to a single service. In addition, in these environments, there are usually several services which are providing the same functionalities with different non-functional parameters, such as quality of service values. Thus, service selection, which deals with choosing from the services with the same functionalities but different quality guarantees, is a challenge in service oriented environments. The current approaches perform service composition and selection on the fly. The services' specifications are retrieved from service registries, which are repositories in which the service providers advertise themselves. As provenance information provide functional and non-functional service specifications, it is a suitable source of data for service composition and selection.

1.6 Contributions and Thesis Overview

This thesis makes several contributions towards addressing current workflow challenges using provenance information. These contributions mainly include:

1. A Provenance Architecture Addressing Workflow and Service Challenges

   As mentioned, current challenges with workflows and services are being addressed using various information sources. For example, assessing workflows is performed by workflow monitoring systems. Mining workflow structure methods exploit workflow event logs generated by some workflow systems. Workflow composition techniques use service repositories to retrieve service specifications. We argue that provenance data provides the required information for all these problems and can be used to target all these challenges while maintaining data consistency. In Chapter 2, a provenance architecture is proposed which addresses the issues with workflows and services. The architecture is composed of 5 components for service composition and selection, workflow structure mining, workflow refinement and evaluation. The requirements for such an architecture along with the methods that can be exploited for each component are discussed. For certain components, a novel approach is proposed and briefly discussed. The work in this chapter has been published in [Naseri and Ludwig, 2010].

2. Workflow Trust Evaluation Using Provenance Information

   As discussed, provenance is also a suitable source for performing evaluations on data. In terms of workflows and services, the evaluation consists of QoS and trust measurements. This is an important and less attended issue in the area of workflows. Workflows need to be assessed and analysed to discover how trustful the composition of services is, therefore, in case the trust provided by a workflow is not satisfactory, the workflow sequence can be either repaired or improved.

As the provenance data is recorded at regular intervals, and consists of values and events that are changing with time, we believe that time series mining methods [Hamilton, 1994] are a suitable choice for evaluating and describing the changes that occur in data with the passage of time to identify the points in time at which a noticeable change in trust occurs. This information can help us to identify which parts of the workflow are not providing the promised or required level of trust. Just like workflows, the services are evaluated. The large fluctuations of the QoS values of services are investigated to predict when in the future the service will not provide the promised QoS. The evaluations are based on statistical approaches, to evaluate the trust of the workflow, and also time series data mining methods, to discover the trend of trust in workflows or services over time.

In case a workflow does not provide the required trust level, or it can not be executed due to a lack of available services, the workflow needs to be repaired or refined. The extracted policy graph of the workflow along with the assessment results of the evaluation component can be used to refine the workflow. The policy graph is traced to find a path that can replace the defective part of the workflow.

In Chapter 3, the workflow evaluation component is targeted. We propose a new approach based on Hidden Markov Models (HMM) [Rabiner and Juang, 1986] for this purpose. The HMM probabilities are learnt using the provenance information. The method is assessed through a case study along with the experimental results. This chapter has been published as a book chapter in [Naseri and Ludwig, 2012].

3. Mining Workflows from Different Perspectives Using Provenance

Discovering workflow patterns has been previously studied using event logs, which provide a very small amount of data for learning the workflow models, while provenance provides a rich knowledge base for extracting hidden and unknown models. Learning and mining workflow patterns and policy graphs, representing the workflow policy, from provenance data is another interesting application of provenance data. Workflow mining discusses techniques for acquiring a workflow model from a workflow log. The process is usually done using an algorithmic technique and/or statistical analysis. Usually machine learning, data mining and workflow approaches are exploited for this purpose. As discussed earlier, workflows can be investigated from many perspectives: functional, behavioral, informational, organizational and operational. In case of the behavioral perspective, which looks at control flow, workflow mining is done by following the order in which events for tasks are stored; for the informational perspective which looks for data flow, usually inputs/outputs stored at the start of event logs are being used; in case of the organizational perspective, participants of tasks and their roles are being discovered in workflow mining. The control flow patterns, which are being discovered present direct, conditional, concurrent and sequential dependencies. Previously, several research studies have been done to discover control flow patterns using event logs. Process mining algorithms can extract the order of the execution of the activities and construct the process model. But many systems are not aware of the tasks of the processes, but instead have knowledge about the documents and the changes made to them. In case of these systems, document versioning logs are often used to mine processes. The process of deriving process models automatically from on-going executions of processes is referred to as incremental workflow mining [Braun, 2006]. This type of mining has the advantage of automatic adaptation in case of changes in processes. The approaches taken are usually semi-automatic and first activity mining from versioning logs is completed, later reverse engineering to derive the overall process model and then the transformation from the system internal model to the external model is performed. The model works incrementally and when new process instances are executed, new records are added to the versioning log and the process model is refined. Current techniques only consider the behavioural perspective (control flow). They assume that the elements of the organizational frameworks are known in advance. On the other hand, they do not address complex iteration constructs, dynamic changes, exceptions, and noisy data, and this is required to investigate mining algorithms and to target these issues. These issues can be targeted using provenance information.

In Chapter 4, we propose a new method for workflow structure learning component which exploits Bayesian structure learning algorithms. Our approach not only considers the behavioral perspective of workflows but also uses the data flow information. Two algorithms of Parents and Children [Spirtes et al., 2000] and Max Min Parents and Children [Tsamardinos and Brown, 2006] are selected for the purpose of this component. These algorithms are modified to support more efficient workflow structure learning. Experiments with different case studies and structural constructs were performed and the results are discussed. This chapter has been published in [Naseri and Ludwig, 2013a].

4. Service Composition and Selection Using Provenance Data in Dynamic Service Environments

   Provenance information can also provide valuable information towards service composition and selection. Service composition is concerned with synthesizing a specification of how to coordinate the component services to fulfill the client request. QoS in Web services encompasses various non-functional issues such as performance, availability, and security, etc. As more services become available, QoS becomes a decisive factor for selecting services. Composition and selection of services requires information regarding services' specifications and their quality values, which are all provided by provenance data. In addition, a history of previous workflow runs provides knowledge towards more intelligent composition. Thus, Chapter 5 revolves around exploiting provenance towards workflow composition and selection. Using of a Partially Observable Markov Decision Process (POMDP) [Murphy, 1982] is the approach we are proposing for this component. It has the advantage of composing and selecting services in one algorithm while providing decision making facilities under the condition of partial observability of services. This chapter was published in [Naseri and Ludwig, 2013b].

5. Conclusion and Future Direction

   In the final chapter of this thesis, we discuss the conclusion and future directions of this research. We outline our achievements through this research and provide information on how this research work can be expanded and improved in the future.

CHAPTER 2

A MULTI-FUNCTIONAL ARCHITECTURE ADDRESSING WORKFLOW AND SERVICE CHALLENGES USING PROVENANCE DATA

As discussed in Chapter 1, in service-oriented environments, services are put together in the form of a workflow with the aim of distributed problem solving. Keeping track of the workflow process along with the data transformations and services provides a rich amount of information for later reasoning. This information, which is referred to as provenance, is of great importance and has found its way through areas in computer science such as: bioinformatics, database, social, and sensor networks, etc. Current exploitation and application of provenance data is very limited as provenance systems started being developed for specific applications. Therefore, there is a need for a multi-functional architecture, which would be application-independent and could be deployed regardless of the application area. In this chapter, we present an architecture which exploits provenance information to target the current challenges of workflows and services in service oriented environments. These challenges include workflow composition, abstract workflow selection, refinement, evaluation, and graph model extraction. The proposed multi-functional architecture addresses these issues by accomplishing reasoning, data mining, and evaluation on provenance data. The requirements for such an architecture along with its building components are discussed. Furthermore, the responsibility of the components, related work and the proposed implementation of each component are presented. This chapter has been published in [Naseri and Ludwig, 2010].

2.1 Introduction

A workflow is defined as the automation of the processes and involves the orchestration of a set of services, agents and actors that must be combined together to solve a problem or define a new service. Different services of the workflow represent the transformation processes that receive the data as input to produce the transformed data as output. The workflow graph often describes a network where the nodes are services and the directed edges represent messages or data streams that channel work or information between services. Each node processes a stream of messages and forwards the resulting streams into its connected nodes.

In service-oriented environments, great numbers of workflows are executed to perform mostly scientific experiments and business tasks.As the number of workflow result datasets increases, it becomes difficult to identify and keep track of them. In addition, in these large-scale scientific computations, how a result dataset is derived is of great importance as it specifies the amount of reliability that can be attributed to the results. Thus, information on data collection, data usage and computational outcome of these workflows provide a rich source of information.

Capturing the execution details of these transformations is a significant advantage of using workflows. The execution details of a workflow, referred to as provenance information, is usually traced automatically and stored in provenance stores. Provenance data contains the information recorded by a workflow engine during a workflow execution. It identifies what data is passed between services, which services are involved, and how results are eventually generated for particular sets of input values. Data associated with a particular service, recorded by the service itself or its provider, is also stored as provenance information. For instance, such data may relate to the accuracy of results a service produces, the number of times a given service has been invoked, or the types of other services that have made use of it.

The stored provenance data is queried and retrieved later for different purposes. This information enables users to trace how a particular result has been arrived at by identifying the individual or aggregation of service(s) that produces such a particular output. The exploitation of provenance data is so limited in comparison to the efforts accomplished and the costs paid for gathering and storing this data [Altintas, 2008]. The most major applications of provenance can be summarized into trust assessment, workflow re-execution and validation, and workflow reduction. A brief introduction of the most common applications of provenance can be summerized as:

- Assessing trust measurements and believability of data, workflows and services is the most important application of provenance. The confidence in the workflow steps executed, the trust of each individual service, and the trust of any data being generated or used can be determined by using the information of the past data or previous executions of services and workflow processes. This subject will be further discussed in the next chapter.

- In many different e-Science areas experiments involve many distributed services maintained by different organizations. After finishing an experiment, it is important to verify that the experiment was performed correctly. Validating the data is possible by doing reasoning on provenance data and checking, for example, whether the services still produce the same results and the workflow is still valid.

- A workflow reduction is performed whereby unnecessary tasks in a workflow are pruned away. This is possible when the datasets to be generated by these tasks have been computed previously, and it is more efficient to access them than to re-compute them. Therefore, the workflow can be reduced by checking the provenance data and finding tasks that have been run previously with their results still available and valid.

Although the mentioned applications provide rich and valuable usages of provenance data, more can be done to take advantage of the stored history of the previous executions. The research done in the area of provenance focuses mostly on the phases a provenance component goes through, such as the capturing mechanisms as well as data retrieval, querying and visualization. Little effort has been invested in discovering general applications for provenance.

One of the unexplored applications of provenance is exploiting it for the purpose of learning. A provenance store provides data related to the previous executions of services and workflows which makes it an appropriate data set for learning and knowledge discovery. The provenance data can be explored using data mining and pattern recognition methods to discover the patterns of interest in the data. These patterns can include workflow structures, trend of the workflow trust, etc. As the store provides large amounts of information on previous executions, therefore, probability learning methods can be used to produce the required parameters for the probabilistic decision making processes. As the provenance data is recorded at regular intervals, and consists of values and events that are changing with time, we believe time series mining methods [Last and Kandel, 2004] are a suitable choice for evaluating and describing the changes that occur in data within the passage of time.

We believe, applying learning and knowledge discovery methods to provenance data can provide rich and useful information on workflows and services. Therefore, the challenges with workflows and services will be studied in this chapter to discover the possibilities and benefits of providing solutions by using provenance data. Previously, a large amount of research has been done to target workflow challenges such as composition, pattern discovery, service selection, and process refinement. In this chapter, an architecture is presented which addresses these issues by exploiting provenance data. The specific contribution of the proposed architecture is its novelty in providing a solid basis for taking advantage of the previous executions of services and workflows along with artificial intelligence and knowledge management techniques to resolve the major challenges regarding workflows. The solution provided for each component is based on data mining methods, time series solutions, and probabilistic decision making processes. The following sections of this chapter are organized as follows: in Chapter 2.2, the mentioned issues along with the motivation and requirements for such an architecture is discussed; in Chapter 2.3, the architecture is presented, along with explanation of its components, Chapter 2.4 presents the related works, Chapter 2.5 provides the implementation methods that can be applied to the architecture; and in the final section, the conclusion is presented.

2.2 Motivation and Requirements

A service-oriented architecture provides an environment in which services are shared among distributed systems. Potentially, thousands of services are available, which can be discovered or combined dynamically through appropriate mechanisms for the purpose of workflow selection, composition, or refinement. Thus, current major issues regarding workflow and services can be summarized in service composition and selection, workflow model extraction, refinement, and evaluation. In previous work, these problems are targeted via semantic descriptions of services and event logs [van der Aalst et al., 2004]. In this section, we are discuss the knowledge requirements of each problem, and will argue how provenance data satisfies these requirements and provides a suitable platform for improving as well as optimizing the quality of the solutions to these problems.

Workflow composition and selection methods require an expressive language that supports flexible descriptions of models and data to facilitate reasoning and automatic discovery and composition. Therefore, they mostly exploit the semantic descriptions of services as well as their QoS specifications from service repositories or service providers to perform the composition or selection. In [Gil, 2005], the authors discuss the requirements for workflow composition. These requirements can be summarized as follows:

- Workflows must be described at different levels of abstraction that support varying degrees of reuse and adaptation. It is important to mention that this requirement is based on the fact that workflows can often be created by reusing existing workflows with minimal changes.

- Expressive descriptions of workflow components are needed to enable workflow systems to reason about how alternative components are related, the data requirements and products for each component, and any interacting constraints among them.

- Flexible workflow composition approaches are needed that accept partial workflow specifications from users and automatically transform them into executable workflows with reasonable levels of certainty.

In order to satisfy these requirements, the authors consider three stages for the creation of the workflows, which include: defining workflow templates, creating workflow instances that are execution independent (abstract workflows), and creating executable workflows (concrete workflows).

We believe the three requirements mentioned above can be satisfied through provenance data. In [et al., 2007], the authors argue that a robust provenance trace provides multiple layered presentation of provenance. A layered architecture and engine for automatically generating and managing workflow provenance data is considered in provenance systems and can be used for interpreting the services and datasets of the workflows. Provenance creation in such an architecture is performed by following a layered approach which fulfills the requirements of the workflow composition process. The first layer of the architecture represents an abstract description of the workflow, which consists of abstract activities with the relationships that exist among them. The second layer provides an instance of the abstract model by presenting bindings and instances of the activities. The third layer captures provenance of the execution of the workflow, including specification of services and run-time parameters. The final level captures execution time specific parameters, including information about the internal state of the activities, machines used for running, status and execution time of the activities.

As the execution time specific parameters are also gathered in provenance stores, provenance data also includes the QoS specifications of services. Thus, service selection solutions can be applied to this data in order to automatically select appropriate services that provide some QoS requirements. Service providers may not be trustworthy enough to deliver the services based on the agreed-on QoS. On the other hand, the "Validity period" of the agreement, which is the duration in which the service provider agrees to provide certain QoS values to the service consumer, might have come to an end and no agreement updates might have been made afterwards. The QoS specification of service providers are described and presented in ontology languages and are stored in service registries. These specifications are updated periodically. In case the QoS guarantees change during a period, the providers will not be able to satisfy the agreed-on thresholds for the service requests which are made before the agreement updates. Using the history of previous executions overcomes the inconsistencies between the guaranteed and delivered QoS values of services to some extent by providing an estimate of the QoS parameters of the services with regards to time.

Most research on workflow systems focus on prediction, tracking and monitoring of workflows, and not on trend analysis or trust evaluation of the workflow processes. The few works which studied evaluation, accomplished a very narrow research goal aimed at improving the performance or fault tolerance of workflow systems [Aiello, 2004], [Truong and Fahringer, 2005]. As the provenance information maintains the records of previous execution details of workflows, it provides the facility to analyze, assess, and evaluate the behavior of a workflow as well as its performance. The performance of a workflow, its believability, improvements, and its future trends, etc. can be analyzed and evaluated through provenance data.

Workflow mining discusses techniques for acquiring a workflow model from a workflow log. The workflow mining methods use the event-logs for discovering the patterns and mining the workflows. Compared to provenance data, event logs keep track of a small amount of information, including mostly service names and execution time. The information provided in event logs is not enough for mining workflows with regards to all the mentioned workflow perspectives, while much stronger reasoning and mining can be done over the data presented in workflow provenance as they provide service input/output information, QoS values, annotations, etc.

To improve the efficiency of the composition and selection processes, previous executions of workflows and services can be used to augment these processes with more intelligence for service composition or selection. Information is learnt from previous executions so that the future service compositions (or selections) disregard the services that either do not have available resources, or do not satisfy the promised trust levels at a particular time. In case of the service composition, the history of previous runs of the workflow processes can be analyzed to discover the possibilities that a certain workflow composition structure would fail during executions.

As more provenance information is gathered through time, the currently in-use workflow process models is refined over time and the structure is geared to improve the efficiency with regards to updates in provenance data. These variations include updates of the most frequently chosen paths, or assigning/changing the weights of the links in the workflow model with regards to their rate of usage in time. These types of augmentations in the model also facilitate the process of refining or repairing a workflow model.

The provenance information might be used to reduce a workflow process by exploiting the information available in provenance store about service outputs. To refine a workflow and replace some parts of the process which can not be executed or do not provide much efficiency, provenance data will be searched in order to discover a more optimal path for the workflow model.

The history of previous executions of workflows and services satisfies the requirements of addressing the discussed challenges. Apart from the requirements, provenance data can address the challenges with more intelligence, efficiency, and reliability. Thus, there is an opportunity for an architecture that facilitates addressing and solving all these issues by exploiting the provenance information.

2.3 Multi-Functional Provenance Architecture

In this section, the multi-functional architecture is presented along with its components.

Figure 2.1 shows the overall view of the architecture. The structure is composed of 5 components that cooperate together along with the provenance store to provide different functionalities.

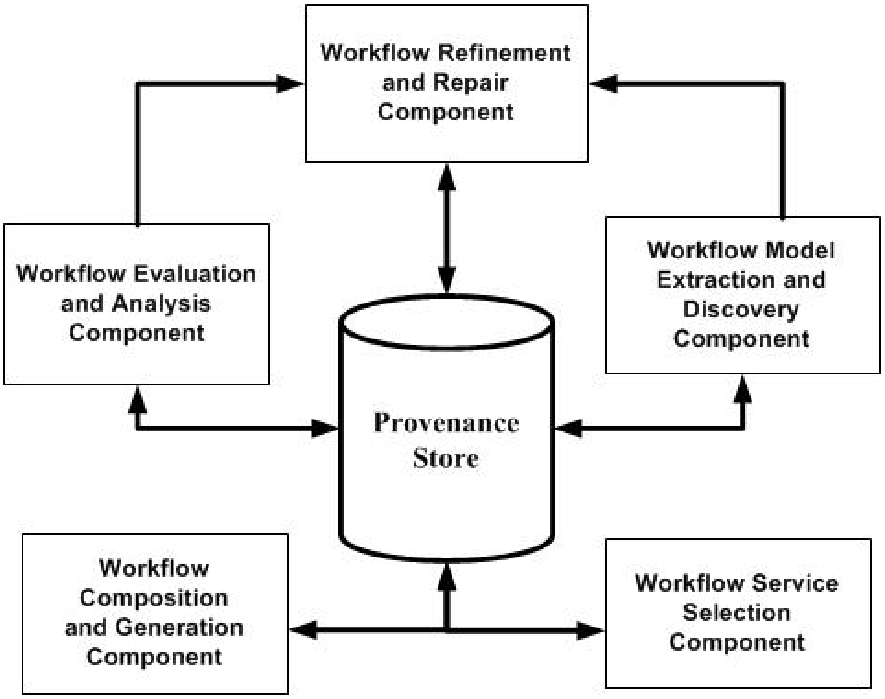

Workflow Refinement and Repair Component
Workflow Evaluation and Analysis Component
Workflow Model Extraction and Discovery Component
Provenance Store
Workflow Composition and Generation Component
Workflow Service Selection Component

Figure 2.1: Multi-Functional Provenance Architecture.

The responsibilities of each component, the way components collaborate to provide the promised functionalities, and the approach taken to achieve the goals of the components are discussed:

1. Workflow Model Extraction and Discovery Component:

   This component is responsible for extracting the workflow pattern and associations that exist among the relevant workflows previously run and executed. Two workflows are considered relevant if they are in the same area of interest. In a workflow management systems, workflows from different types of areas are executed which may contain no service overlaps. In order for two workflows to be considered relevant, they should share some similar services or have similar service connections in their worklfow model. Their workflow models might contain the same abstract services but they have used different implementations of those services. The extraction component discovers the hidden connections that might exist among services which have not been known beforehand. It generates a graph model of the relevant services, with edges representing the associations between them. These associations can be of data flow, or control flow types or combination of both. The output is a policy graph including all observed paths that could exist between the relevant services that belong to a certain area. The extracted policy graph can be used later for the purpose of workflow construction and repair. The component is also able to receive a workflow pattern, and look for the same pattern sequence in the store to discover if there is any information regarding its previous executions in the provenance store.

2. Workflow and Service Evaluation Component:

   Evaluating workflows and services in terms of trust and quality is an important and less studied issue in the area of workflows. Workflows need to be assessed and analyzed to discover how trustful the composition of services are, therefore, in case the trust given by a workflow is not satisfactory, the workflow sequence can be repaired and improved. Another responsibility of this component is to identify the points in time at which a significant variation in trust occurs. This information can help us in identifying the parts of the workflow that are not providing the promised or required trust. Similar to workflows, the services are evaluated by this component. Large fluctuations of the QoS values of services are investigated to predict when in the future the service will not support the promised QoS. Based on the previous executions, this component is also able to predict which services are potentially going to be executed and in case the results of another instance of the same service are available, the process of workflow execution can be possibly improved by exploiting those results. Apart from the trust assessment, the performance of the workflow is evaluated in terms of the resource usage, and total time elapsed from the submission to completion.

3. Workflow Repair and Refinement Component:

   In case a workflow does not satisfy the trust level requirements specified by the user, it can not be trusted and needs to be refined. In addition, due to lack of available services for a workflow, it can not be executed and needs to be repaired. The repairment/refinement component takes advantage of the extracted policy network of the workflow model extraction component along with the assessment results of the evaluation component. The extracted network is traced to find a path that can replace the defective part of the workflow. The defective path is either inefficient due to lack of trust provision, or can not be executed any longer because of unavailable services. In case the evaluation component predicts that a service will not provide its promised nonfunctional requirements, the workflow repair and refinement component is responsible for replacing this service by another service or services that provide similar functionalities but satisfy the promised nonfunctional requirements.

4. Workflow Composition and Generation Component:

   Composing a set of services using provenance data is a very useful exploitation of the provenance store. The stored specifications of services and their states, which evolve through time, can help support of composing the services automatically. On the other hand, having the previous history of executions, provides the data, i.e., service specification data, which is essential for learning the workflow composition. Therefore, the composition will be done in a more intelligent way by exploiting the provenance data. This component receives the requirements and composes a workflow dynamically by taking advantage of the service specifications provided in the store. Previous execution of workflows enables the composition to be more robust as it exploits the evaluation results of services and workflows to generate a well-designed workflow process.

5. Workflow Service Selection Component:

   The problem of selecting a set of concrete services that provide the required QoS specifications for a complete abstract workflow is referred to as abstract workflow service selection problem. The provenance data can be exploited to speed up this task. In order to find the set of concrete services that match a single abstract service, service registries are looked at and matchmaking algorithms are applied to discover matching services. The service discovery phase is much simpler if provenance data is used. Previous executions of workflows along with the workflow templates simplify the process of service discovery for a simple query. The set of suitable concrete services for the abstract workflow can then be selected more optimally by using the selection mechanisms along with the evaluations of previous executions.

2.4 Related Works

The workflow model and policy graph extraction component can be implemented using different methods and algorithms. Discovering models of processes, and mining sequences, are all relevant areas and the techniques being applied for these purposes can be used for the case of workflow model extraction. Therefore, methods and solutions exploited in these areas were studied. The current solutions include data mining methods for discovering sequential patterns, statistical analysis methods for building and extracting statistical dependencies, or a combination of both methods. In [Altintas, 2008], the authors discuss the data mining algorithms to discover sequential patterns. The algorithms include the Generalized Sequential Pattern (GSP) algorithm, which has the advantage of taking the time constraints into account, and Apriori algorithm [Agrawal and Srikant, 1994]. Methods used for event-data analysis are a set of techniques which are used for process discovery. These methods vary from purely algorithmic ones to purely statistical ones [Hwang and Yang, 2002] or a combination [Gaaloul et al., 2005]. In [Gaaloul and Godart, 2005], the authors propose an algorithm which assumes an interval for the execution of an activity instance. The resulting extracted graph contains the control dependencies and conditions which are discovered by the algorithm. Some of the methods used for discovering sequences and processes were previously exploited by the research done in the area of workflow pattern discovery from event logs. In [Aalst and Dongen, 2002], techniques were developed for discovering workflow models from timed logs. In [Huang and Chang, 2008], the workflow patterns are discovered by mining frequent episodes.

As will be described in the following, many research efforts address the problem of workflow composition. Theorem proving methods are used which describe the available services and user requirements in a first-order language, and generate constructive service ordering proofs with theorem proving. Service composition descriptions are then extracted from particular proofs [Waldinger, 2000a]. The work in [Narayanan and McIlraith, 2002], presents a logic programming language built on top of the situation calculus. The web service composition problem is then addressed through the provision of high-level generic procedures and customizable constraints. One of the most studied areas of workflow composition is solving the problem via AI planning techniques. The state change produced by the execution of the service is specified through the precondition and effect properties which are provided in the semantic service descriptions [Wu et al., 2003]. A high-level declarative description has been used in some works to achieve service composition through rule-based planning. The method uses composability rules to determine whether two services are composable [Medjahed et al., 2003a].

In the case of abstract workflow selection, several works [Sannella, 1994], [Forman, 2003] have addressed this issue, proposing exact algorithms or heuristics to determine the appropriate concrete services for each individual component invocation or over the complete composite request. [Ardagna and Pernici, 2005] maps the service selection for workflows into a Multi-Objective Programming model (MOP). [Berbner et al., 2006] models this problem as an optimization one and adopts a genetic algorithm for solving it.

2.5 Methodology

In this section, we discuss our proposed approaches towards implementation of the architecture. Chapters 3-5 will describe our methodologies and experimental results for these components in more detail.

The architecture implementation is mostly based on artificial intelligence and statistical methods. Current workflow model discovery methods are not able to discover the workflows from different perspectives. In addition, not many of the current solutions can discover the parallel sections of a workflow structure. Our process discovery method for workflow model extraction is based on Bayesian reasoning. The method used by this component exploits the Bayesian structure discovery technique "to learn" the workflow model and build the workflow graph. In order to model the problem as Bayesian structure discovery, the services serve as the nodes of the Bayesian network, each having values representing service names. The links in the resulting graph represent the causal relationships that exist among the services. Therefore, the graph extracted from the provenance data depicts the workflow policy graph.

The evaluation component is based on statistical approaches such as HMM [Rabiner and Juang, 1986] and multivariate time series methods [Hamilton, 1994]. These solutions are used for analyzing and evaluating the trust of the workflow, or to discover the trend of trust in workflow or services over time. This component evaluates the trust of a workflow using a HMM and specifies the trends of changes in workflow trust over time. As the non-functional specifications of services, such as execution time, are also gathered through provenance systems, QoS parameters of services can be considered as time series data. Therefore, the time series evaluation method is applied on the data to provide assessments for trust and QoS values of workflows and services. Event detection methods can be exploited to find the trend of services and predict when in future the services might not be able to provide the promised values.

Our solution to the composition problem is based on POMDP [Murphy, 1982] techniques. POMDPs provide a suitable approach for composing services. A discrete POMDP models the relationship between an agent and its environment. The parameters of the POMDP – which include conditional transition probabilities, conditional observation probabilities, and rewards– are learnt through the data available in the provenance store. The planning process is augmented with learning methods to make the composition as intelligent as possible.

The service selection component of the architecture will use the HMM [Rabiner and Juang, 1986] sensor scheduling approach [Guralnik and Srivastava, 1999]. Sensor scheduling is the problem of optimally choosing which single sensor to use at each time instance to minimize a cost function. Past observations together with past choices of sensors affect which sensor to choose at present. This problem perfectly matches the abstract workflow service selection. The sensors to be chosen at each time represent the concrete service that should be chosen at that time instance. The solution to the sensor scheduling problem selects an appropriate concrete service at each time instance while keeping the total values of QoS specifications as low as requested. The sensor

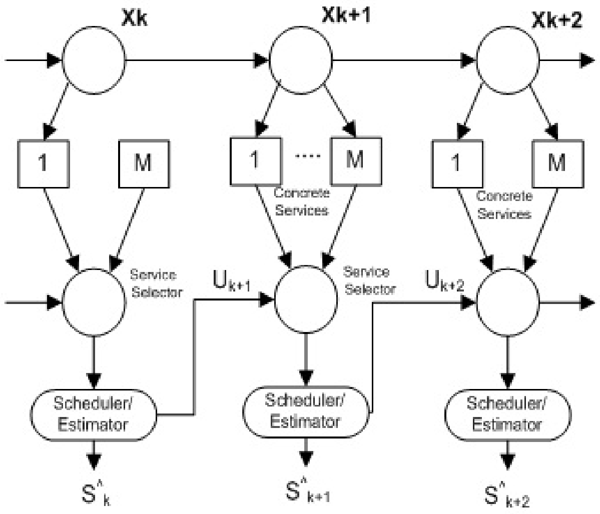


Figure 2.2: HMM for Service Selection.

scheduling problem proceeds in three stages for each time instance. The first phase, which is the scheduling, determines the sensor that is to be used at the next time step given the information available at the current time, which includes sensors chosen at previous times as well as the observations measured by the sensors. The next stage evaluates the observation of the sensor chosen for the next time step. The final stage computes the optimal state estimate by using the HMM state filter. Each states with regards the to service selection problem is represented by the state of each concrete service. Figure 2.2 shows the HMM [Rabiner and Juang, 1986] sensor scheduling model.

2.6 Evaluation Design

In order to evaluate the architecture's performance, different provenance systems were studied to investigate the one which best satisfies the data requirements for the components. Taverna [Tav, 2013], Triana [Tri, 2003], and PASOA [Moreau et al., 2013] are the provenance systems studied. Triana does not provide a separate provenance system, instead, it has a rudimentary history tracking system that allows workflows to be stored with the interim states of the components in the workflow. This annotated workflow can then be replayed to generate the same results. Taverna is a workflow workbench that has a provenance model which captures both internal provenance locally generated in Taverna and external provenance gathered from data providers. The provenance data gathered in Triana is very limited in comparison to Taverna and does not support annotations. Although the PASOA project presents an architecture which addresses issues such as provenance generation, representation and reasoning, its implementation is not complete and is just intended as a technology preview. In order to perform real world and valuable experiments with the architecture, Taverna was selected as a practical provenance system and was expanded to incorporate the additional features of the proposed architecture.

The evaluations that will be considered for the architecture include evaluating the accuracy and performance of the workflow model extraction component with regard to the graph provided. The refinement component will be assessed to observe the rate of improvements of the workflow. The behaviours of the components will be assessed in terms of scalability to observe the effect of different number of services on the model. The performance of the components will be compared with on the fly solutions to investigate the influence of learning as well as the feedbacks fed into the components from previous executions.

2.7 Conclusion

In this chapter, a multi-functional architecture was proposed which addresses the current issues of workflows and services using provenance data. The components of the architecture, and the proposed implementation methods were briefly introduced. It can be seen that the provenance data can provide a richer knowledge base for workflow pattern discovery compared to event logs; therefore, more inference can be done for extracting the workflow patterns. In case of service composition, the provenance data can work as the semantic repository and by applying the learning methods on the previous executions, a more efficient composition is achieved. Furthermore, little research has been done on workflow evaluation and assessment, while, having the provenance information, different types of analysis and reasoning can be performed on processes and services. More evaluation along with stronger and more intelligent reasoning leads to better results. The evaluation component improves service operations by providing feedback to services about its behaviour, and in cases in which the QoS values provided by a service violates its promises in terms of QoS guarantees. In addition, workflow assessment results enable comparing similar workflows with each other in terms of resource usage, trust guarantee, and speed. These results are used in workflow composition, service selection, and workflow refinement components and augment the architecture with intelligence and robustness.

The different techniques applied to the same problem will be compared with each other in terms of their execution time, support, and scalability. The proposed architecture will be augmented with other services to provide more functionality, robustness, and reliability. Components will return feedback to the provenance store to augment the data with information learnt. As a result, the stored data will be updated dynamically through time and annotations will be added to the information. Thus, the components will operate more intelligently. The provenance data will be preprocessed, cleaned, and the possibilities of unknown, missing as well as erroneous data will be considered.

CHAPTER 3

EVALUATING WORKFLOW TRUST USING HIDDEN MARKOV MODELING AND PROVENANCE DATA

In service-oriented environments, services provide different qualities in terms of parameters like availability, cost, reputation, execution time, etc. A trust score can be derived from these QoS parameters, which determines the rate of reliability in each service. This score can assist the service consumer parties to decide whether or not to transact with that service provider in the future. In such distributed environments, services with different functionalities are combined together to define new services or provide higher level functionalities. Having a trust score for each service, the trust level of a combination of services, i.e. a workflow, can be determined. Assessing the trust value of a workflow helps to determine its rate of reliability. Therefore, the trustworthiness of the results of a workflow will be inferred to decide whether the workflow's trust rate should be improved. The improvement can be done by replacing services with low trust levels with services with higher trust levels. We provide a new approach for evaluating workflow trust based on an HMM. We first present how the workflow trust evaluation can be modeled as an HMM and provide information on how the model and its associated probabilities can be assessed. Then, we investigate the behavior of our model by relaxing the stationary assumption of HMM and present another model based on non-stationary HMMs. We compare the results of the two models and present our conclusions. This chapter has been published in [Naseri and Ludwig, 2012].

3.1 Introduction

A workflow is defined as the automation of the processes and involves the orchestration of a set of services, agents and actors that must be combined together to solve a problem or define a new service. Different services of the workflow represent the transformation processes that receive the data as input to produce the transformed data as output. The workflow graph often describes a network where the nodes are services and the edges represent messages or data streams that channel work or information between services. Each node processes a stream of messages and forwards the resulting streams into its connected nodes. The workflow activities are run repeatedly by one or more users and large amounts of resulting provenance data are produced. Capturing the execution details of the workflow transformations is very beneficial. They identify what data is passed between services, which services are involved, and how results are eventually generated for particular sets of input values. Such data may also relate to the accuracy of results a service produces, the number of times a given service has been invoked, etc. [Naseri and Ludwig, 2010]. The provenance data can be explored using learning methods to discover the patterns of interest in the data.

A description of QoS specifications as well as well-defined inputs and outputs is usually presented in the service ontologies provided in service registries. As the provenance store keeps the specification of services such as input or output or service description, it can be regarded as a large informational registry, providing the chance of workflow performance analysis using previous experiences. Applying learning and knowledge discovery methods to provenance data can provide rich and useful information on workflows and services. Among the workflow issues and challenges, workflow analysis and evaluation, which mostly includes QoS assessment and trust measurements, is the least-attended problem. Provenance provides a suitable resource of information for performing analytical evaluation. Thus, in this Chapter, we focus on this component and present how workflow trust can be assessed using provenance information.

Execution of a sequence of services requires much more resources and time in comparison to a single service. Thus, if a workflow is not very reliable, many resources and time will be wasted, since the results of the workflow can not be trusted. Therefore, it is important to be able to evaluate the trust of a workflow to find the degree of reliability of the workflow and its results. This also helps to decide whether the workflow needs some refinement and whether less trustworthy services should be exchanged with more trustworthy ones. Having the trust value of each service allows evaluation of the overall trust value of a sequence of services, i.e., a workflow. Therefore, we can determine the amount of trust that can be placed on the overall workflow as well as the results and datasets generated during the workflow execution.

The remaining sections of this chapter are organized as follows: Chapter 3.2 outlines how workflow trust can be evaluated using HMM; in Chapter 3.3, we discuss the procedure followed for assessing the HMM probabilities, and in Chapter 3.4 the implementation details of the model are provided. Chapter 3.5 presents a case study, as well, the stationary assumption of the model is investigated and some experiments are performed to compare the Non-Stationary HMM (NSHMM) trust evaluation results with HMM. In the final section, the conclusion and future work are given.

3.2 Related Works

There are very few approaches addressing the subject of workflow trust evaluation. One approach uses a decision tree model, which is presented in [Rajbhandari et al., 2006], in which a decision tree is built out of a question sequence that will help in assessing the trust that can be associated with the data produced from a process. The root node asks about the trust of the workflow and has three child nodes, evaluating the trustfulness of services, data and the workflow process. Each child node has a sub-tree, representing a set of yes/no questions. The decision making process starts with one child node, traverses its sub-trees and continues to the next child node. This procedure is followed continuously until all the sub-trees are investigated. The result of the investigation is either a "yes" or "no", determining whether the workflow can be trusted or not. This work has been extended and an important shortcoming of it, the crisp result, has been addressed in [Rajbhandari et al., 2008]. Therefore, the outcome of each analysis node of the trust decision tree is mapped to a fuzzy membership function. Later, these values are combined together using fuzzy inference rules.

However, all the current solutions lack accuracy, automation, and reliability. They are based on a decision tree model with categorical nodes that have been designed by the developers. The decision nodes of the tree are simple sets of questions regarding the user's views or behaviors toward service, data or process trust. Besides, the trust value of each service or data is not considered separately, but instead the overall trust level of services is involved in the decision making process.

We propose a new approach for the evaluation of trust of workflows, which is based on a statistical model named HMM [Rabiner and Juang, 1986]. Rather than traversing a set of question nodes, in our model, the trust will be assessed by solving a set of mathematical equations that describe the behavior of the workflow trust in terms of random variables and their probability distributions. Thus, our method is more accurate in comparison to the previous approaches and will support automation.

Many approaches have been proposed to improve the predictive power of HMM in practice. For example, a factorial HMM [JingHui et al., 2005] is proposed to decompose the hidden state representation into multiple independent Markov chains. In speech recognition, factorial HMMs can help in representing the combination of multiple signals. An hierarchical HMM [Fine et al., 1989] is another method that facilitates the inference of correlated observations over long periods in the observation sequence via higher level hierarchy. However, from the essential definition of HMM, there are other ways to improve the predictive power of HMMs. One approach is to relax the stationary hypothesis of HMMs and make use of time information. This method is referred to as Non-stationary Hidden Markov Models (NSHMM) [B. Sin, 2008]. To investigate this research further and observe the behavior of our model with regards to the non-stationary assumption, the workflow trust has also been evaluated using NSHMM.

3.3 Hidden Markov Modeling for the Evaluation of Workflow Trust

In probability theory, a stochastic process [Karlin and Taylor, 1975] is a collection of random variables used to represent the evolution of some system over time. In a stochastic process, there is some indeterminacy which results in several directions in which the process may evolve. A Markov process [Karlin and Taylor, 1975] is a stochastic process that satisfies the Markov property which states that the conditional probability distribution of future states of the process depends only upon the present state, not on the sequence of events that preceded it. A model that assumes the Markov property is referred to as a Markov Model [Karlin and Taylor, 1975]. An HMM is a statistical Markov model in which the system being modeled is assumed to be a Markov process with unobserved , i.e., hidden, states. In an HMM, the state is not directly observable, but the observation, which is dependent on the state, can be observed. Each state has a probability distribution over the

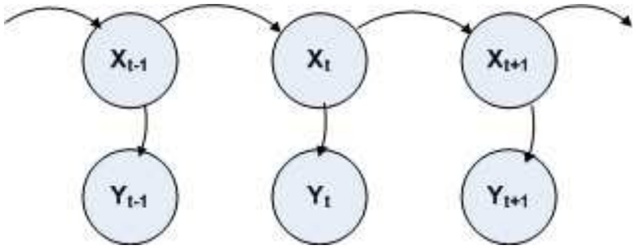


Figure 3.1: Basic HMM.

possible observations. Thus, the sequence of observations generated by an HMM provides information about the sequence of states. Given an HMM, and a sequence of observations, the probability of the observation sequence given the model can be evaluated. It is also possible to discover the hidden state sequence that was most likely to have produced the observation sequence. Another type of inference on HMMs can estimate the HMM model through training examples and learning methods.

The HMM model basically consists of two finite sets of variables: state variables and evidence variables, which are also called the observations. The state variables are the hidden variables that change over time; while the evidence variables are the observable variables that are known in advance at each time step. The challenge is to determine the hidden parameters from the observed ones.

Figure 3.1 shows a simple first order HMM. The state variable x_t is a hidden variable at time t and can have a value from the domain of x_t, i.e., $x_t \in x_1, x_2, ..x_n$, where n is the number of states. The random variable y_t denotes the observable parameter at time t. From the figure, it can be seen that the value of the hidden variable at time t, i.e. x_t, depends only on the value of the hidden variable x_{t-1}, and other previous parameters have no influence on it. This property is referred to as the first order Markov property.

HMM has become the method of choice for modeling stochastic processes and sequences in applications such as speech and handwriting recognition [Rabiner, 1989], computational molecular biology [Krogh et al., 1994], natural language modeling [Jelinek, 1985], etc. In this work, HMM is used for the purpose of workflow trust evaluation.

In order to be able to assess the proposed HMM model, provenance information are exploited. Provenance is one of the growing demands in distributed service oriented environments, which supports the systems with documentation of the origin and the processing steps of data that is part of a workflow execution process. It also provides explanations about which, how and what resources and services were used to produce that data, and is referred to as provenance data that is captured and stored in provenance stores for the purposes of reasoning, validation and re-execution. A provenance store provides the necessary information that is exploited for the purpose of estimating HMM probabilities.

It is important to mention that in a great many applications of HMMs, the latent states cannot be observed. As a result, based on the observations, the probabilities involved are assessed using the Maximum Likelihood (ML) method. But in case of our model, we are assuming that the provenance information permits accurate enough information concerning the latent states to directly estimate the transition probabilities using such data.

3.4 Methodology

The notion of trust of an enacted workflow is an important issue in distributed service oriented environments. Trust evaluation aims at contributing in the discovery of how trustworthy the results of a workflow are. It also helps the optimization of composite service executions. In this section, we are going to first present how the workflow trust can be evaluated using hidden Markov modeling. Later, we explain how the model can be assessed by taking advantage of the previous history of the execution of workflows.

In our multi-functional architecture which was discussed in the previous chapter, we mentioned that workflows are composed and services are selected using the service composition and selection components. The goal of the service selection component is to find a service from the list of available services, such that the users end-to-end QoS requirements are satisfied. Service selection mechanisms are based on the prediction of services performance from the quality advertised by providers as well as a service's reputation. The selection accuracy of the service selection mechanisms are improved by exploiting service trust values along with the QoS parameters. Thus, during the workflow composition and selection processes, services are selected based on their QoS values as well as the trust values. As a result, in order to model the workflow trust evaluation as an HMM, the state and observable variables are mapped as follows:

- Tr_t: the trust state variable, represents the state of the trust of the workflow at time t. The trust state is a continuous scalar quantity which holds a value within the range $[0-1]$. A trust value of 1 presents a fully trustworthy service, while a value of 0 declares that the service can not be trusted at all. For the sake of simplicity, the trust state variable used in experiments in this research is considered a discrete variable of the domain set: Low, Medium, High.

- S_t: the evidence variable represents the service that is being executed at time t. As for the evidence variables, QoS values of the services are taken into account. The evidence variable, i.e. service, is a scalar discrete quantity.

Figure 3.2 depicts a simple linear workflow and the corresponding HMM, modeled to evaluate

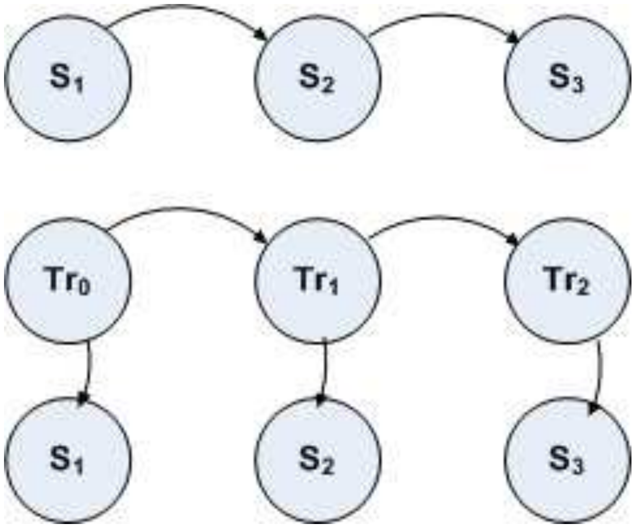


Figure 3.2: A sample workflow and the HMM for workflow trust evaluation.

the trust level of the workflow. As can be observed from the figure, the state of the trust of the workflow at each time step, will be determined by investigating the state of the workflow trust at the previous time step, and observing the service that was executed at that time.

In the theory of HMMs, some assumptions are made for the sake of mathematical and computational tractability. Here we present how these assumptions can be applied to our model:

1. The Markov assumption: It is assumed that the next state is dependent only upon the current state. This is true in case of our model, as the state of the trust of the workflow at each time only depends on the state of the trust at the previous time and not the other prior states.

2. The output independence assumption: This is the assumption that the current observation is statistically conditionally independent of the previous observations. In case of our model, the QoS values at time *t* are conditionally independent of the previous services given the current state.

3. The stationary assumption: This assumption is based on the fact that the transition proba-

bilities between the states are independent of the actual time at which the transitions take place. In case of the workflow trust problem, we cannot say that transition probabilities are completely independent of time. We suppose that this assumption will be true for our model since we can take the average of the state transitions of all times and have one set of state transition probabilities for the overall time period. In order to investigate this further, later in this chapter, we will observe the behaviour of the model by relaxing this assumption and having a non-stationary HMM.

Having defined the HMM and described how the HMM parameters and assumptions can be mapped to the workflow trust evaluation parameters, we will now clarify how this model can be exploited for the purpose of trust evaluation.

As mentioned earlier, different kinds of inference can be done on HMM structures. These include methods for computing the posterior distribution over the current, future, or a past state, or finding the sequence of states that is most likely to have generated those observations. The posterior distribution is the distribution of an unknown quantity, the state of the the trust of the workflow, conditional on the evidence obtained, i.e., the service observed. As for the word "Posterior" in this context, it means after taking into account the relevant service related to the particular case being examined.

Filtering or monitoring is the task of computing the posterior distribution over the current state, given all evidences and observations to date. The following probability expresses filtering inference:

$$P(X_t \mid y_1, y_2, ..., y_t) \tag{3.1}$$

Using the filtering model, the probability of the state of the trust at the final state of the workflow can be roughly estimated given all the observations, which are the services seen so far. Therefore, for the case of the trust evaluation, the following probability should be assessed:

$$P(Tr_2 \mid s_1, s_2, s_3) \tag{3.2}$$

for different possible trust state levels. Evaluation of the above probability provides us with estimations of probabilities for different trust levels at time t_2. As mentioned above, in this work, the state of the trust will be evaluated at three different levels of *High*, *Medium* and *Low*. The work can later be extended to support further trust levels as well as continuous workflow states.

3.4.1 Trust Model Assessment

In order to be able to compute the filtering inference, two other probabilities should be assessed beforehand. These probabilities are referred to as state transition probability and sensor probability. The state transition probability is defined as the probability of being in the next state given the current state, i.e. $P(x_t \mid x_{t-1})$, which in our case is the probability of being at a trust level at time t given the level at the previous time, i.e. $t-1$. The sensor probability is defined as the probability of the observation at time t, which is the service that was executed at time t, given the different level of trustworthiness of the workflow at that time. To assess the state transition or sensor probabilities, provenance data is being used.

It was mentioned earlier in Chapter 1 that in scientific computations using workflows how a result dataset is derived is of great importance as it can specify the amount of reliability that can be placed on the results. Thus, capturing provenance information is very advantageous in this regards. Provenance information facilitates data dependency determination, workflow result validation, efficient workflow re-executions, error recovery, etc. [Altintas, 2008]. In this chapter, we are exploiting the provenance data to evaluate the trust of the workflows by learning the HMM probabilities.

Information of the previous executions of a certain workflow, provide us with the data about workflow trust changes in history. We will be using this data to assess the HMM for the workflow at run time.

Assessment of Transition Probabilities

In order to assess the transition probabilities, the trust state transitions, i.e. $P(Tr_t \mid Tr_{t-1})$, should be computed for all pairs of workflow services that are being executed in sequence. Having a large provenance record of the previous executions of workflows, we will be able to learn the transition probabilities from the data.

To assess this probability, we determine the number of each trust state transition with regard to the total number of transitions of that state. The transition probability estimation for our model is computed based on Equation 3.3:

$$P(Tr_t = j \mid Tr_{t-1} = i) = \frac{n_{ij}}{n_i} \tag{3.3}$$

where n_{ij} denotes the number of transitions from trust level i to trust level j, and n_i denotes the number of transitions from trust level i. These values are calculated through provenance information using previous executions of services. For example, for the sample workflow in Figure 3.2, which was composed of three services, the trust state transition from high to low will be computed by first determining the number of high to low transitions for the service pairs (s_1, s_2) on transition from s_1 to s_2 and dividing it by the number of times the service s_1 had high trust level. The same will be done for the pair (s_2, s_3). The average of these values represents the transition probability from high to low.

It is important to mention that the same pair of sequential services might be repeated in several workflows, and the transition probabilities for these services will be learnt without considering specific workflows. The average of all these probabilities will denote the final transition probability for these pairs of services.

Assessment of Sensor Probabilities

To assess the sensor probabilities for each time instance t, the probability of observing an evidence variable given the state at that time should be computed. Therefore, we should compute $P(S_t \mid Tr_t)$, which again will be learnt from the provenance data.

For this purpose, the number of times the trust state of service instance S_t was at each trust level is estimated. This value is divided by the total number of times any service was at that trust state. As before, the provenance history of the workflow will be used. Equation 3.4 represents the assessment of the sensor probabilities for our model:

$$P(S_t = s_t \mid Tr_t = j) = \frac{n_{stj}}{n_j} \tag{3.4}$$

where n_{stj} denotes the number of times being in state j and observing service s_t, and n_j denotes the number of times a transition from state j has occurred.

Assessing the Trust Level

The filtering method, also refered to as the forward algorithm, is used to calculate the probability of a state at a certain time, given the history of evidence. To assess the workflow trust at a time that all the services in a sequential workflow have been run and observed, we apply the filtering method. Having assessed the sensor and transition probabilities, we will be able to assess the filtering model of HMM and therefore evaluate the workflow trust using Equation 3.5:

$$\begin{aligned} P(Tr_t \mid S_1 = s_1, S_2 = s_2, ..., S_t = s_t) \propto P(S_t = s_t \mid Tr_t) \sum_t P(Tr_t \mid Tr_{t-1}) \\ P(Tr_{t-1} \mid S_1 = s_1, S_2 = s_2, ..., S_{t-1} = s_{t-1}) \end{aligned} \tag{3.5}$$

The probability of $P(Tr_{t-1} \mid S_1 = s_1, S_2 = s_2, ..., S_{t-1} = s_{t-1})$ is computed recursively. Equation 3.5 evaluates the probability of different trust levels at time t having observed the services the workflow is composed of until that time.

3.4.2 Cases with Dynamic or Parallel Sections

The presented trust model is compatible for workflows which contain not only sequential but also parallel sections in the workflow. In case of non-sequential workflows, a sequential workflow is extracted from them by selecting a subsection out of all parallel sections according to a policy, and replacing that parallel subsection with the selected subsection. Starting with the parallel subsection with maximum numbers of parallel levels, a subsection is chosen for parallel level by first applying the HMM model to all the parallel sub-sections of that section, and then the trust level probabilities of the sub-sections are compared with each other. For each section, the trust level is combined with the frequency of executions of that section and and the parallel section is replaced by the subsection that provides better results. By following this policy for all the parallel levels, the workflow is transformed to a sequential workflow, and finally the HMM model is applied to assess the trust level.

It is important to mention that as the proposed approach exploits provenance information to get an assessment of the QoS values, it works for the static scenarios.

3.5 Implementation

As mentioned earlier in this work, the trust of each service instance is categorized into three levels of *High*, *Medium*, and *Low* and can be evaluated by aggregating the QoS parameters of the service. These QoS parameters can include status, availability, reliability, execution time, reputation, etc. The trust value is usually determined by assigning a weight to each parameter and the summation of the multiplication of the parameters by their weights results in the final trust value. As in our current model, we are concerned with trust levels rather than trust values, we determine the level of the trust with regard to the level of the QoS parameters.

In our implementation, we have considered the QoS parameters of status, reliability and availability. The QoS parameter *status* is a binary value that represents the status of the execution of the service. A value of 1 describes that the service was executed successfully and a value of 0 reports unsuccessful execution. The QoS parameter availability presents an estimation of availability of a certain service and its data, while reliability denotes the degree we can rely on the processing and the response time of the service. Both parameters have a value in the range of [0,1].

In order to decide about the trust level of each service using these parameters, we followed a table model, Table 3.1, in which the level of all QoS parameters of availability and reliability in conjunction with the status of the execution determines the level of the trust. The table is referred

Table 3.1: Trust level decision table, L, M, and H denote Low, Medium, and High. This table represents how, in our implementation, the combination of several QoS values is mapped to a trust state.

Trust	Reliability Availability Status
L	LL0
L	LL1
L	ML0
M	ML1
L	HL0
M	HL1
L	LM0
M	LM1
L	MM0
M	MM1
L	HM0
H	HM1
L	LH0
M	LH1
M	MH0
H	MH1
M	HH0
H	HH1

to as the trust level decision table throughout this chapter. A sample row in this table represents the associated trust level in combination with the discussed QoS parameters. For example, LL1 denotes that the level of the reliability and availability of a service is *Low*, and the status is 1. According to the table, the trust level of the service is assessed as *Low*.

The levels of *reliability* and *availability* of the services are determined according to a set of predetermined range levels. For the examples and experiments provided in this chapter, the following range table (Table 3.2) was used.

As was discussed earlier, the probabilities are assessed by applying learning methods over the provenance data. For the purpose of learning, we implemented a provenance store in MySQL [MyS, 2013] including tables for storing the information of workflows, services, workflow instances, and

Table 3.2: Range Level of the QoS parameters Availability

Trust Level	**Low**	**Medium**	**High**
Availability	[0,0.3]	(0.3,0.7)	[0.7,1]
Reliability	[0,0.3]	(0.3,0.7)	[0.7,1]

workflow sequences. The provenance data is then generated by a random workflow generator implemented to produce instances of a workflow. While the generated data does not consider all aspects of real world workflows such as missing data, hidden correlations, etc, the approach can be applied to real scenarios with preprocessed data. The generator asks for the following parameters as input:

- N_s: the number of services the workflow should be composed of.

- N_w: the number of previously executed instances of the workflow.

In order to assess the HMM, we followed the algorithm which describes the sensor and transition models in form of matrices. The transition matrix denoted by Γ is a $m \times m$ (in our case 3×3) matrix where m is the number of possible states. The probability of a transition from state i to state j is denoted by the entry Γ_{ij}:

$$\Gamma_{ij} = P(Tr_t = j \mid Tr_{t-1} = i) \tag{3.6}$$

which, as discussed, will be evaluated using the generated provenance data along with the trust level decision table (Table 3.1), QoS parameters range level (Table 3.2).

The sensor model is also put into matrix form. For each time step t, a diagonal matrix of size $n \times m$, O_t, is constructed whose diagonal entries are given by the values $P(S_t \mid Tr_t = i)$, with the other entities set to 0. Assuming to have m numbers of observations,each entry is the probability of the observed event given each state.

Now, to accomplish the filtering inference and represent the forward messaging in HMMs using the matrix model, Equation 3.7 is applied recursively:

$$f_{1:t+1} = \alpha O_{t+1} \Gamma^T f_{1:t} \tag{3.7}$$

where α is the normalization factor and Γ^T represents the matrix transpose of Γ. At each step, this process is carried forward with additional observations. The probability vector that results contains entries indicating the probability of being in each state conditional on the input seen thus far.

3.6 Case Study

In this section, we present a workflow scenario and describe how its trust can be evaluated using the presented model. The sample workflow is the process of knowledge discovery in databases, which is referred to as the KDD process [Fayyad et al., 1996]. The KDD process is composed of four services for data selection and cleaning, data transformation, data mining, and data interpretation. Figure 3.3 shows the process.

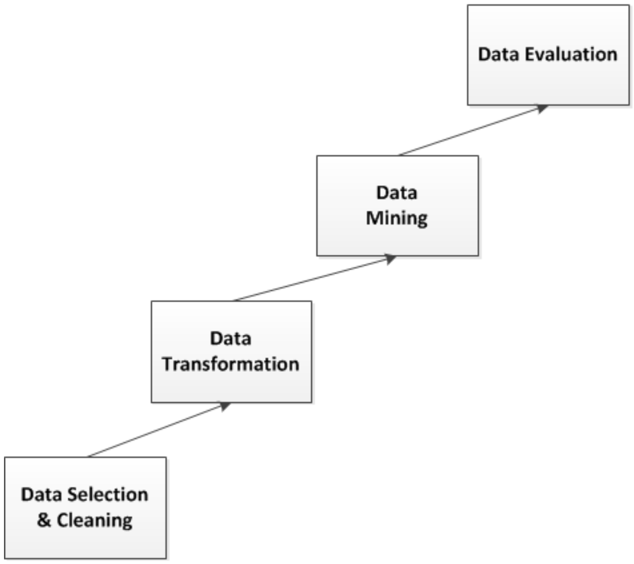


Figure 3.3: A sample workflow scenario - KDD Process.

The following assumption is made. A distributed service-oriented environment is sharing services for the purpose of knowledge discovery, and a workflow is executed using four different services shared by service providers in the environment, each having different QoS values, and therefore, different trust estimations. Using the workflow generator, the above workflow was defined and 50 execution instances were generated, representing the provenance data. The workflow data generator receives the following input parameters: minimum and maximum range for each QoS parameter, the size of the workflow and the number of workflow instances that should be generated. It then creates a sequential workflow of the given size, and produces workflow instances by generating random values between the minimum and maximum ranges for each QoS parameter. Table 3.3 shows the average of the QoS parameters of the instances generated for the case study.

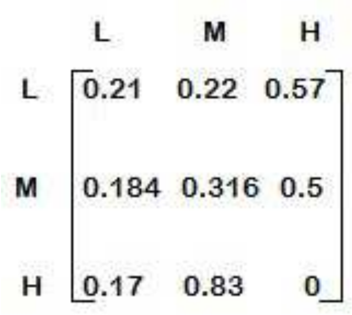
The QoS parameters *availability* and *reliability* were randomly generated in the range of 0.3 to 0.9, which mostly covers the medium and high trust levels. The status of the execution was set to zero in less than 20% of the cases. It is important to emphasize that according to the trust level decision table (Table 3.1), the state of the trust of a service instance is evaluated as *Low* if its status

Table 3.3: The average of the values of the QoS parameters generated for the scenario.

QoS Parameter	Reliability	Availability	Status
Data Selection	0.58	0.59	0.8
Data Transformation	0.7	0.7	0.88
Data mining	0.34	0.34	0.82
Interpretation	0.84	0.84	0.82

is zero. The reason for this decision is that if a service does not complete its execution successfully, that service instance should not be trusted at all. Therefore, we evaluate the trust as low regardless of the instance's level of *reliability* and *availability*.

In the next step, the transition matrix is built by learning the probabilities from the generated provenance data. Given the data, the transition matrix T, of the above example was estimated as given in Figure 3.4.



$$\begin{array}{c|ccc} & L & M & H \\ \hline L & 0.21 & 0.22 & 0.57 \\ M & 0.184 & 0.316 & 0.5 \\ H & 0.17 & 0.83 & 0 \end{array}$$

Figure 3.4: Transition matrix of the example

L, M, and H represent the trust levels *Low*, *Medium* and *High*. An entry Γ_{ij} denotes the transition probability of being transferred from trust level i to j. For a better understanding, the state transition diagram is also provided in Figure 3.5, which is the same as the transition matrix but presents it in a graphical view which is easier to follow.

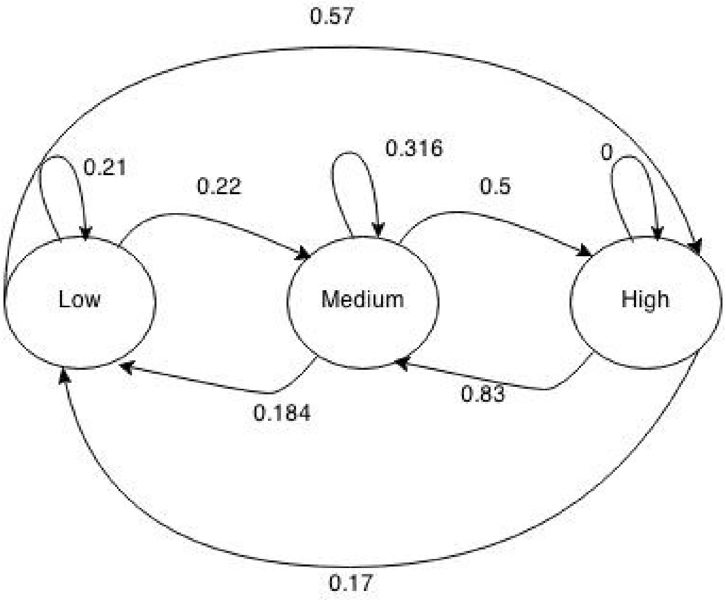


Figure 3.5: The state transition diagram showing the transition probabilities for the above example learnt.

Having learnt the transition matrix using the provenance information, the forward algorithm starts with assessing the sensor probability at the first time step and forwards this information along with the transition messages to the next time step. This process of forwarding messages continues until the last service is observed, and therefore the overall trust of the workflow is evaluated. It is important to mention that the prior belief about the trust state probabilities, i.e., the initial state probabilities, is considered equal for all the three possible states and was set to 0.33 for all the trust levels.

To investigate the behavior of the filtering method and observe the workflow trust level estimations along time, the reader can refer to Figure 3.6. The figure shows how the trust state of the workflow changes over time during the HMM assessment for the discussed example.

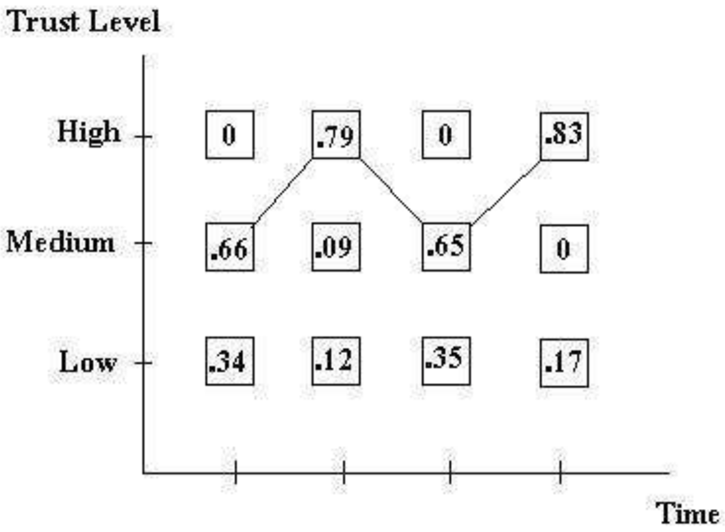


Figure 3.6: The change of the trust state probabilities over time using an HMM.

It can be observed that using the filtering algorithm, the workflow trust state is evaluated as *Medium* after observing the first service, it then heads toward *High*, then again *Medium* and finally the workflow trust level is evaluated as *High*.

Taking a look at the average values of the QoS parameters of each service explains the behaviour of the model. According to the QoS range evaluation 9 (Table 3.3), the trust level of the first service, which is the data selection service, can be evaluated as *Medium*. The trust level of the third service is also evaluated as *Medium*, and the trust level of the second and the fourth service is estimated as *High*.

The explanation above and the transition matrix shown in Figure 3.4 describe the reason behind the path taken in Figure 3.6. The path shows the route between the trust levels with the highest probabilities at each time step. The transition probabilities with large probability values include transitions from *High* to *Medium*, *Low* to *High*, and *Medium* to *High*. The evaluation process starts with the first service which has an average of *Medium* trust level. As the transition probability of *Medium* to *High* is the largest, this leads the state of the trust toward *High*. Being in state *High* and having observed a service with *High* trust level leads the trust level toward *Medium* as the largest transition probability from *High* is the one toward *Medium*. The rest of the transitions can be explained in the same way.

It should be considered that there is always a less than 20% probability for a low trust state to be chosen for all the services. Because as discussed earlier, the status of the executions of services were randomly set to zero in almost 10 to 20 percent of the cases. Therefore, the final trust level probabilities will have a 10% low level probability on average.

3.6.1 Investigation of the Stationary Assumption

It was mentioned earlier that one of the assumptions of HMM is the stationary assumption. In order to follow this assumption, the transition probabilities were assessed by taking the average of the transitions between each pair of services to have the same state transition matrix at all times. This can not be assumed to be true in case of the workflow trust problem. As a wokflow process is composed of services which are connected and executed along the time, the workflow trust state transitions are time-dependent. This section investigates how the model will behave if we relax this assumption and transition probabilities are considered time-dependent. To achieve this goal, the transition probabilities are computed separately for each time step. It is important to mention that when a workflow is sent to a workflow management system, the sequence is known in advance. Thus, the time dependency of services is given. However, the runtime workflow parameters, such as the QoS values, are determined at run time.

In the theory of HMMs, it is assumed that state transition probabilities are independent of the actual time at which the transitions take place. This assumption can be mathematically presented as:

$$P(x_{t_1+1} = j \mid x_{t_1} = i) = P(x_{t_2+1} = j \mid x_{t_2} = i) \tag{3.8}$$

for any t_1 and t_2. Equation 3.8 states that the transition probabilities are constant over time which means that the probability of transition between different trust levels is the same for all times. Therefore, the Markov chain is described as *stationary* in the strictest sense. In general, it is possible to lift the constancy constraint and define the transition probabilities as a function of time. This model is referred to as the Non-Stationary Markov Model (NSMM) [Bongkee and Jin, 1995] and has a set of transition probability distributions that vary over time. This means that, given a state i, the probability of moving to another state j in the next time step is a function of time. The time can be either absolute or relative. Equation 3.9 shows how the state transition function can be estimated:

$$P_{ijt} = \frac{C(i,j,t)}{C(i,t)} \tag{3.9}$$

where $C(i,j,t)$ is the co-occurrence frequency of state i and state j at time t and it can be estimated by counting the co-occurrence times of state i and state j at the t^{th} time. $C(i,t)$ is the frequency of state i at time t and can be estimated by counting the occurrence times of state i in the t^{th} time. And P_{ijt} is the transition probability between state i and j at time t.

In case of the workflow trust evaluation, the trust state of a workflow process is affected by the services which are the building blocks of the workflow. In a workflow graph, these services are connected to each other and are executed in an specified order along time. Thus, the transition probabilities can be considered as a function of time since the probability of transition from one trust level to the other at time t depends on the services that are being executed at that time instance. Therefore, it is important to investigate the behavior of the model this time using the NSHMM in order to observe the effect of the stationary assumption on the trust evaluation results.

In case of relaxing the stationary assumption for the workflow trust evaluation, the state transition probabilities are assessed separately at each time step and a transition matrix is built using the provenance data representing the history of the observations seen previously at those time steps.

The transition probability from state i to state j at time t will be assessed as follows:

$$Pt(Tr_t = j \mid Tr_{t-1} = i) = \frac{n_{ijt}}{n_{it}} \tag{3.10}$$

where n_{ijt} denotes the number of transitions from trust level i to trust level j at time t, and n_{it} denotes the number of transitions from trust level i at time t.

The non-stationary model was further implemented and the result of the same scenario studied in the previous section was investigated using the new model. It is observed that the trust state probabilities have not changed much as time elapses. The maximum trust level path follows the same routine with very little changes in the state probabilities at each time. The evaluation result of the NSHMM shows that the workflow can be trusted with a probability of 93%, while using the HMM this probability was 83%.

To investigate this further, we ran experiments using both models and compared their results. The experiments were done by creating workflows of sequential structures with 5 to 25 services in increments of 5. The workflows were created using the discussed workflow generator and workflow instances were produced. A previous execution history of 100 instances was randomly generated for each workflow in order to learn the sensor and transition probabilities. For both HMM and NSHMM models, the filtering algorithm was run for each workflow size. The average of the resulting workflow trust level probabilities was then computed for each workflow instance. It was observed from the experiment results that for both models the distance between the same trust levels was equal in 96% of the cases.

Figure 3.7 represents the average resulting trust level probabilities of the HMM and NSHMM, assessed through the filtering method. It can be observed that the differences are very small. In all the experiments, the level of the trust was estimated to be the same.

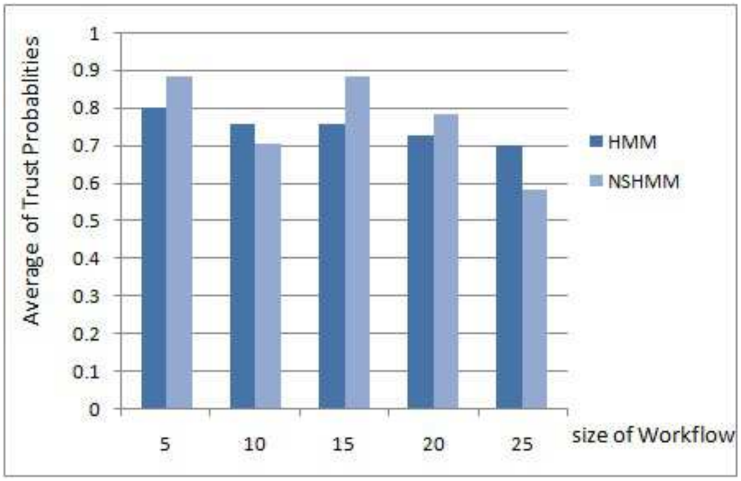


Figure 3.7: Comparing the average trust level of HMM vs. NSHMM for 5 to 25 numbers of services with increments of 5.

In order to determine whether the results of the two models are the same, we ran the paired t-test on the datasets of the two models. For each model, a data column is generated such that each value in the column represents the average value of the resulting highest trust level probabilities assessed for all the workflow instances of a certain size. The t-test is a statistical test that assesses whether the means of two groups of data are statistically different from each other. The result was a p-value of 0.78, which represents that the datasets are not significantly different from each other. The chart in Figure 3.7 and the t-test results both verify that the stationary assumption does not have a significant effect on the results of the trust level assessment, as both models provide estimations for the same trust levels with very little difference.

Experiments were done to compare both models in terms of the execution time and it was observed that while there is not large differences between the execution times, the execution time of the non-stationary model is larger. The reason for this observation goes back to the transition matrices that should be computed for each time instance separately while for the HMM with stationary assumption, the transition matrix is built once.

3.7 Conclusion

In this chapter, a multi-functional architecture was described that addresses the current research issues of workflows and services using provenance data. The components of the architecture were described consisting of model extraction and discovery, workflow evaluation, workflow repair and refinement, workflow composition, and workflow service selection.

In addition, we focused on one component of the multi-functional architecture and put forward an approach for evaluating workflow trust level using hidden Markov models and provenance data. We discussed how the HMM assumptions can be applied to this problem, and we provided details on how the model can be assessed using the provenance data.

In order to investigate the behaviour of the model, we provided a workflow scenario and expressed how its trust level is evaluated using the proposed model. In order to verify the effect of the stationary assumption of HMMs for the trust evaluation problem, we investigated the results of applying the non-stationary hidden Markov model to our problem.

The two models were then compared with each other. It was observed that the same trust level was estimated by both models with a small difference in their probability values. Therefore, the stationary assumption does not have a significant impact on the trust evaluation results. The non-stationary assumption of transition probabilities seems to be more accurate in case of our model since the probability of moving from one state to the other at a time instance depends on the state of the two services that are being executed at those times. Thus, for this problem, it is better to consider the transition probabilities as time-dependent probabilities for more accurate results.

Future work involves performing a large number of experiments to evaluate the scalability and accuracy of the system, preferably with real data. Various experiments will be done for different workflow sizes, and the behavior of the system could be observed in response to larger workflows.

As the amount of provenance data affects the accuracy of the learnt probabilities, the reliability of the system will be evaluated considering different learning data. The HMM parameters can be learnt using the ML or EM mthods to improve the accuracy of the method provided.

The main concern of the current implementation was randomly generating a large amount of valid provenance data for many workflows, each having similar structure with the others. The future workflows ought to be realistic and consist of common services and patterns with reasonable provenance values and data from a number of executions. The model will be improved to also consider trust values of the workflow process and input data for the evaluations.

Furthermore, the fluctuation of trust with the Markov process needs to be investigated in order to discover the points at which the workflow lacks trustworthiness and should be refined. It is desired to automatically detect and replace less trustworthy services with trustworthy ones. This part of the work can be extended by learning the workflow patterns from the provenance data and substituting less trustful services or sections of the workflow with more trustworthy ones.

CHAPTER 4

EXTRACTING WORKFLOW STRUCTURES THROUGH BAYESIAN LEARNING AND PROVENANCE DATA

In this chapter, we investigate the workflow model extraction component. Mining workflow models has been a problem of interest in literature for the past few years. Event logs have been the main source of data for the mining process. Previous workflow mining approaches mostly focused on mining control flows that were based on data mining methods, and exploited time constraints of events to discover the workflow models. Using provenance information, we present a mining approach which not only takes the behaviourial aspect of workflows into account, but also considers their informational aspect. Thus, the resulting structure displays not only the control flow information but also the data flow information. Provenance information is a proper source of reasoning, learning, and analysis for this purpose since it provides information regarding the service inputs, outputs and quality of service values. Therefore, provenance data along with Bayesian structure-learning methods are exploited for process mining in this chapter. Two constraint-based Bayesian structure-learning algorithms are investigated and modified to overcome the constraints that might be implied in certain workflow models. The experiments show that the modifications lead to better mining results on three common mining scenarios. This chapter has been published in [Naseri and Ludwig, 2013a].

4.1 Introduction

Combining a set of tasks together in a specific order for the purpose of achieving a specific goal is a process taking place in all different areas of science, from business to chemistry, physics, math, etc. During such a process, referred to as a workflow process, tasks, well-defined steps in a workflow model, might have prerequisites and are run in sequential or parallel orders.

The most common form of representing a workflow model is the directed graph. Tasks or activities are usually enclosed in boxes or circles and are referred to as vertices of the graph while the arrows depict the edges, which represent the direction of the flow. As for some workflow systems, the workflow events and their timing information are recorded sequentially into logs. Keeping track of certain data attributes of a process being executed, and storing this information into an event log is a procedure taking place in certain systems. These event logs usually contain a limited amount of information about the process and mostly include the process id, the name of the task, and the execution time of each task. Mining workflows and processes through event logs has been a problem of interest in the literature. Event logs have been analyzed and searched in order to analyze the effectiveness of a workflow process, to discover previous workflow models, to find the hidden causal relationships existing among tasks, etc [Agrawal et al., 1993; Aalst and Dongen, 2002].

The process of workflow mining is referred to as the task of extracting process knowledge from the event logs. It discusses techniques for acquiring a workflow model from a workflow log. As mentioned in [Tiwari et al., 2008], the desire for companies to learn about their processes is the main reason behind exploitation and development of process mining techniques.

The basic idea behind workflow mining is to construct the workflow's directed graph from the information gathered through the workflow process's run. This process is usually conducted using an algorithmic technique or statistical analysis. Machine learning, data mining, genetic algorithms, and sequence mining are the main approaches of workflow mining applied in the literature. Data mining methods for discovering sequential patterns, statistical analysis methods for building and extracting statistical dependencies, or a combination of both methods have been used.

As described in [van der Aalst and Weijters, 2004], some challenging problems usually discussed in the area of workflow mining include mining loops, incomplete data, and mining workflows from different perspectives. In addition, the current methods do not address complex iteration constructs, dynamic changes, and noisy data. As workflows can be investigated from different perspectives, the mining process can focus on functional, behavioural, informational, organizational or operational aspects of a workflow. It was briefly mentioned in Chapter 1 that the behavioural perspective looks at the control flow and workflow mining inspects the order in which events for tasks are stored. The control flow patterns discovered through process mining present direct, conditional, concurrent and sequential dependencies. The informational perspective looks for the data flow and exploits the inputs/outputs stored at the start of event logs for this purpose. In case of the organizational perspective, participants of tasks and their roles are being discovered through the workflow mining process.

In this chapter, we exploit Bayesian structure learning methods along with provenance information to mine workflows from different perspectives. Bayesian learning benefits from great amounts of data that is provided by provenance information. To use the combination of informational and behavioural data, service outputs along with start and execution times of services are exploited for workflow structure learning. The constraint-based Bayesian learning algorithms that will be presented, use service output values to learn the causal relationships existing between services through evaluation of their information. Two constraint-based algorithms of Parents and Children (PC) [Spirtes et al., 2000] and Max-Min Parents and Children (MMPC) [Tsamardinos and Brown, 2006] were selected for learning, and modified in order to improve the workflow mining task. Our approach is different from the previous ones as it exploits both data and control aspects of workflows for mining, discovers concurrent processes, and supports structures with duplicate tasks.

The rest of this chapter is organized as follows: in Chapter 4.2 related works are discussed, our methodology is described in Chapter 4.3 along with the constraints, conditions, and modifications applied to the constraint-based algorithms. Chapter 4.4 provides the implementation details. In Chapter 4.5, a case study is conducted showing three different cases, and the performance evaluation results are presented. Chapter 4.6 presents the conclusion of the research provided.

4.2 Related Works

Data mining algorithms of Apriori basis [Gaaloul et al., 2005] have been used in literature to discover sequential patterns. Methods used for event-data analysis, from purely algorithmic ones [Hwang and Yang, 2002] to purely statistical ones [Hwang and Yang, 2002], or a combination of both techniques [Gaaloul et al., 2005] have been applied to this problem.

In [Aalst and Dongen, 2002], techniques were developed for discovering workflow models from timed logs. The model presented is based on Petri Nets [Murata, 1989] and mining methods are provided for discovering the transactions that occurred between tasks which rely on detecting the causality. In [Huang and Chang, 2008], the workflow patterns are discovered by mining frequent episodes and a statistical dependency table is constructed. The resulting extracted graph contains the control dependencies and conditions held between the tasks.

In [Agrawal et al., 1993], the sequential or concurrent nature of joins and splits of workflow patterns are used for the purpose of finding the workflow model. The workflow mining technique presented follows the three steps of constructing a statistical dependency table, discovering frequent patterns, and mining patterns by combining the previous two steps. The workflow is discovered by finding joins, sequential and fork patterns from the information stored in the event database.

Deriving process models automatically from on-going executions of processes is referred to as incremental workflow mining [Braun, 2006]. This type of mining has the advantage of automatic adaptation in case of changes in processes and workflows. The approaches taken are usually semi-automatic. At first, activities are mined from the versioning logs, and afterwards, a reverse engineering process is performed in order to derive the overall process model. The research provided in [Tiwari et al., 2008] presents a comparison of the current workflow mining approaches and categorizes them based on the contribution of each work. Table 4.1 presents the approaches and their contribution towards workflow mining. Some mining techniques, such as the one presented in [Herbst and Karagiannis, 2000], are limited to sequential models. Other approaches, such as [Schimm, 2004] and [deMedeiros et al., 2004], support more complex structures, including concurrent processes, but are limited to workflow models without repetitive or duplicate tasks. In [Agrawal et al., 1998], [Herbst and Karagiannis, 2004], [van der Aalst and de Medeiros, 2005], and [Dongen and van der Aalst, 2005], approaches are presented that allow the appearance of the same task in the workflow model. Some methods, such as the Markovian approach [Doshi et al., 2005], do not target any of the major workflow mining issues.

As the table presents, current approaches do not mine process models from different perspectives. Also, there is no one single approach that targets many of the workflow mining issues. Thus, new approaches are required that address these issues.

Table 4.1: Comparison of some workflow mining approaches [Tiwari et al., 2008].

	Data mining based	**Genetic algorithm**	**Other approaches**
Duplicate tasks	[Agrawal et al., 1998], [Herbst and Karagiannis, 2004]	[van der Aalst et al., 2005]	[Dongen and van der Aalst, 2005], our approach
Different perspectives			Our approach
Heterogeneous data sources			[Dongen and van der Aalst, 2005]
Concurrent Processes	[Schimm, 2004]	[van der Aalst et al., 2005]	[deMedeiros et al., 2004], our approach
Process Rediscovery			[van der Aalst et al., 2002]

We believe that these challenges can be resolved if the logs provide more information. As discussed in [van der Aalst and Weijters, 2004], a process data warehouse is required to apply workflow mining. One of the components of our proposed architecture is the workflow policy graph extractor, which learns and mines workflow patterns from provenance data. Following the proposed architecture, in order to be able to mine workflow models using various perspectives and to simplify the discovery of causal relations, we propose the exploitation of provenance information for the purpose of process mining.

Provenance data is a suitable source of information for process mining since it provides vast amounts of information about previous runs of services and workflows. This information ranges from service specifications and Quality of Service (QoS) values to data and control flows generated during workflow execution runs. The input/output specifications and values recorded by the provenance system during workflow executions facilitate mining workflows using the informational aspect. QoS parameters of time, availability, etc. enable applying time series methods for the purpose of control flow mining.

As mentioned earlier, thus far, most research efforts have focused on the control-flow perspective of workflows. The approach we are taking in this thesis, exploits data flow along with control flow information to extract workflow models from provenance data. In order to discover workflow models from data, the process variables and values are used along with service names and timing information. The result is a process model that not only incorporates the control flow dependencies, but also the informational dependencies of services.

4.3 Methodology

In this section, we discuss our methodology. Constraint-based Bayesian structure learning method is introduced and the workflow extraction problem is modeled using this method. Later, we present Bayesian structure learning algorithms we exploited through this thesis along with the modifications applied to them to improve their efficiency for our purpose.

4.3.1 Bayesian Structure Learning Methods

In order to extract control flow relations along with the data flow dependencies existing between services, the workflow process discovery method proposed in this thesis is based on Bayesian learning approaches. A Bayesian network is a probabilistic graphical model that represents a set of random variables and their conditional dependencies via a directed acyclic graph (DAG). Directional relationships in a DAG represent the cause-effect relationships in such networks. As discussed in [Margaritis, 2003], learning Bayesian Networks is being used for inferring possible causal relations. Many of the independencies in a domain can be presented through a Bayesian Network structure.

A probability distribution assigns a probability to each measurable subset of the possible outcomes of a random experiment of statistical inference [Wik, 2011]. The following condition declares the intuitions which connect the causal graphs with probability distributions [Spirtes et al., 2000]:

Definition 1: Let P be the probability distribution. The causal structure represented by graph G generated by G and P satisfies the Markov Condition if and only if
$\forall v \subset V, v\, is\, independent\, of\, V \setminus (Dec(v) \cup Pa(v))\, given\, Pa(V)$ where V denotes the set of graph vertices, $Dec(v)$ represents the descendants of node v and $Pa(v)$ represents the parents of node v.

Basically, the Bayesian structure learning methods can be categorized into the two groups of constraint and score-based algorithms. Constraint-based algorithms perform structure learning in two steps. The first step discovers the skeleton using conditional hypothesis tests. The skeleton is the undirected structure in which only the location of edges are determined with no directions. The conditional independence can be defined as follows:

Definition 2: In a probability distribution P, two variables of X and Y are conditionally independent, $Ind(X,Y|Z)$, given variable Z if the following condition is satisfied: $P(X,Y|Z) = P(X|Z)P(Y|Z)$. This condition states that X and Y are conditionally independent given Z if and only if, given any value of Z, the probability distribution of X is the same for all values of Y and the probability distribution of Y is the same for all values of X.

The second step finds the orientation of the edges in the skeleton. The scored-based methods address structure learning as a model selection problem. After having defined a scoring function that evaluates how well a structure matches the data, these methods search through all possible network structures for the highest scored network, which results in an NP-hard (Non-deterministic Polynomial-time hard) approach. Compared to score-based methods, constraint-based approaches are more suitable for the purpose of knowledge discovery as they produce more accurate results.

These methods, which are also referred to as conditional independence learners, use conditional independence tests to detect the Markov blankets of the variables in order to compute the structure of the Bayesian network. The Markov blanket for a node X in a Bayesian network is the set of nodes composed of parents of X, its children, and its children's other parents. Conditional independence tests in structure learning are concerned with nodes/variables that are necessarily independent given the structure of the underlying DAG. The independence assertions are learnt from data and are used in both steps of these algorithms. The first phase exploits the conditional independence test to determine whether an edge should exist between two nodes, and represents the result as an undirected skeleton. In order to learn the structure of DAGs, a sufficient condition is "faithfulness". Given any graph the Markov condition determines a set of independence relations. A probability distribution *P* on a causal graph *G* which satisfies the Markov condition may have independence relations that have not been entailed by the Markov condition. If all the independence relations of *P* are entailed by the Markov condition, it is said that *P* and *G* are faithful to one another. A distribution *P* is said to be faithful if some DAG exists to which it is faithful. The faithfulness assumption asserts that the conditional independences observed in the distribution of a network are due to the structure of the network and are not accidental properties of the distribution. It allows us to move from a probability distribution to a DAG.

4.3.2 Modeling the Workflow Model Extraction Problem as Bayesian Structure Learning

Constraint-based approaches in literature are different based on the type of independence test or ordering heuristics. Among the possible algorithms, we have selected the PC, as well as, the MMPC algorithms. Using statistical or information-theoretic tests, these algorithms estimate based on the data whether certain conditional independencies between the variables hold. They start from a complete, undirected graph and delete edges recursively based on conditional independence decisions. This yields an undirected graph, which can then be partially directed and further extended to represent the underlying DAG. The PC algorithm has an intuitive basis and guarantees the recovery of the original causal structure under ideal conditions [Abellan et al., 2006]. It is faster than similar approaches such as SGS [Daly et al., 2001] and produces better results. The other algorithm, MMPC, outperforms on average several constraint-based algorithms such as PC, Sparse Candidate [Friedman et al., 1999], etc. [Tsamardinos and Brown, 2006]. As both of these algorithms take a similar approach towards structure learning, i.e., using conditional independence tests, they were selected as the main methods applied to the workflow data for the aim of mining.

Constraint-based structure learning methods are based on the assumption of having a very large database. This condition is satisfied since provenance information is being used for this purpose. Provenance data provide us with great numbers of records of previous executions of services and workflows. Thus, enough information on Service's output values, execution time stamp, etc. are available for the purpose of constraint- based learning. The other condition that should be satisfied is the faithfulness condition. Thus, the independence relationships should have a perfect representation of a DAG. In case of workflow mining, these relationships represent the data flow connections that exist between services (tasks). This condition can not be assumed to be true, since these relationships in a workflow structure might not necessarily represent a DAG depending on the degree of "faithfulness" of the data. Thus, we will be presenting modified versions of these algorithms called Modified*PC and Modified* MMPC. These algorithms provide better results in case of unfaithful workflow models.

In order to model the problem workflow mining as a Bayesian structure discovery, services serve as the nodes of the Bayesian graph, each having values representing different states a service provider provides. The links in the Bayesian graph represent the causal relationships that exist among the services. Causation is a relation between particular events. Both the cause and effect of a causal relation are particular events. In a workflow graph, services perform the role of the events in a causal graph. As an example, consider two services of "WhatIsMyIP" and "WhatIsCity". The first service provides the Internet Protocol (IP) information of a certain internet user. The second service, receives an IP address, and provides information about the city, the IP address refers. The relation between these two services presents a causal relation. By applying the structure learning methods on provenance information of services, the graph extracted from the provenance data depicts the workflow policy graph.

To evaluate the degree of dependence between services using conditional independence tests, the output parameters and values of services are taken into account. The values are matched against each other to assess the mutual information the services provide, given the current discovered structure, and investigate if dependencies can be found.

Current structure learning solutions discover the Bayesian model assuming the faithfulness condition is true. In case of our model, we cannot assume to have a Bayesian structure in the data if the workflows do not follow the conditions underlying such a structure. The faithfulness of the data can not be guaranteed as it depends on the workflow models. Thus, the Bayesian learning algorithms need to be modified in order to be able to discover the structure as accurately as possible even in the absence of the faithfulness condition.

Assume a workflow consisting of 3 sequential services of A, B, and C for finding an Internet Protocol (IP) address, searching for the city name based on the IP, and finding the weather forecast for that city. This workflow structure example will not satisfy the faithfullness condition as both "IP" and "city" services' output values provide the same mutual information for the "weather" service. Thus, "weather" and "city" are assessed as independent given "IP". Such issues prevent the Bayesian structure learning algorithms to discover the whole model of a workflow. Given the described example, the learnt model will only include arcs from A to B and to C, and services B and C are assumed to be conditionally independent. This will result in a workflow structure represented as *B <- A ->C* while the original model suggests *A ->B ->C*.

In order to overcome these issues, the original PC and MMPC algorithms are augmented with certain heuristics that make their results more accurate in case of the unfaithfulness condition. These heuristics exploit timing information provided in the provenance data, gathered during workflow executions, to identify the ordering of the variables for variable selection, to discover the mutually exclusive or parallel services, and to find services that provide the same information.

In the following sections, we explain the proposed algorithms of Modified *PC and Modified* MMPC. As mentioned earlier, these algorithms were built upon basic PC and MMPC algorithms and have been changed to support the graph discovery in case the faithfulness condition does not hold for the workflow model and data.

4.3.3 Parents and Children (PC) Algorithm

The first phase of the PC algorithm will be used for learning the structure of the graph. It starts by forming the complete undirected graph. This algorithm thins this graph by omitting the edges based on a conditional independence test, denoted as $Ind(x,y|S)$ in the algorithm. In the beginning, it removes the edges with zero order conditional independence relations. An nth order conditional independence relation includes n variables in the conditional set. In the next step, the first order conditional independence relations are taken into account and so on. The set of variables conditioned on need only be a subset of the set of variables adjacent to one or the other of the variables conditioned, denoted by $Adj(x)$ function. As discussed in [Spirtes et al., 2000], the performance of this phase of the algorithm can be improved by knowing the ordering of the edges. Since the provenance information are used for learning, the timestamps including the starting time of services and duration can be exploited for the purpose of ordering. This results not only in a better performance of the algorithm with less time complexity, but also saves execution time of the second phase of the algorithm that is aimed at discovering edge directions. The Modified_PC algorithm is shown in Algorithm 1. Apart from using a chronologically ordered set of nodes, the two functions of "Check_Splits" and "Check_Same_Info" are the main modifications applied to the original PC algorithm. The first function, i.e., Check_Splits, checks if the two variables being checked for independency are parallel-splits (and-splits or or-splits). A parallel split creates a split in a workflow model. In case of and-split, all the branches will be active, as for an or-split only one branch is active at a time. In order to check this, the starting time of the two variables, i.e., services, are taken into account. If the time difference between the two starting times is less than a threshold, the two services are considered parallel and based on their data values are added to the split-and or split-or lists. The other function, "Check_Same_Info", checks if the two variables x and y being checked for independence given variable z provide the same information, i.e., fall into the same information provider category. It performs the assessment by first checking if x belongs to y's conditional set. If this is not the case, y and z are tested for independence conditioned on x. If either of these tests are true, then x and y and z provide the same information, and thus, the scenarios such as the one presented in the previous subsection are discovered correctly.

Algorithm 1 Our Proposed Modified_PC Algorithm

```
function MODIFIED_PC($G, O$)
    Input: Fully connected graph $G$, Timely Ordered Variables $O$
    $V$: the set of node variables for graph $G$
    $i = 0$
    repeat
        for all $x \in V$ do
            for all $y \in Adj_x$ do
                if $Check\_Splits(x, y, G, O)$ then
                    Continue
                end if
                Determine if $S \subset Adj(x)ny\ with\ |S| = i\ and\ Ind(x, y|S)$
                if this set exists then
                    if $CHECK\_SAME\_INFO(x, y, S)$ then
                        if time_difference(x,y) < time_difference(y,S) then
                            Remove S-y link from $G$
                            break
                        else
                            Remove x-y link from $G$
                            Add y-S link to $G$
                            break
                        end if
                    else
                        Make $S_{xy}$ = S
                        Remove x-y link from $G$
                    end if
                end if
            end for
        end for
    until an $i$ is found for which $|Adj_x|<i\forall x$
end function
function CHECK_SAME_INFO($x, y, cond$)
    if $!check\_mutual(x, cond)$ && $!check\_mutual(y, cond)$ then
        if $x \in Conditional\_Set(y, cond)$ then
            return true
        end if
        if Ind($y, cond|x$) then
            Add $x$ to $Conditional\_Set(y, cond)$
            return true
        end if
    end if
    return false
end function
```

4.3.4 Max-Min Parents and Children (MMPC) Algorithm

This algorithm is based on the local discovery algorithm called Max-Min Parents and Children (MMPC) [Tsamardinos and Brown, 2006]. The Max-Min part of the algorithm name refers to the heuristic the algorithm uses, while the parents and children part refers to its output.

MMPC focuses on learning substructures around each variable. It is invoked by each variable of the network, referred to as t, in order to identify the existence of edges to and from that variable, and to discover the structure of the network. Similarly to PC, this algorithm starts with a fully connected graph and exploits two heuristics to discover the dependencies. The first phase, which is referred to as the forward phase, incrementally discovers edges using the Max-Min heuristic. The Max-Min heuristic selects the variable that maximizes the minimum association with a selected variable relative to the so far learnt graph. It uses the function $\mathrm{Assoc}(x,t|Z)$ which measures the strength of dependency between x and t given a set of variables Z. As mentioned in [Tsamardinos and Brown, 2006], the justification for the Max-Min heuristic is to select the variable that remains dependent even after conditioning all the subsets of the so far discovered network. The second phase removes the false positives that might have been entered in the first phase by running conditional independence tests on x and t given any subset of the learned graph.

The mutual information existing between the output values of the two services is used as the criteria evaluating the strength of the association. The MinAssoc function determines the minimum dependency achieved between x and t over all the subsets of the variables discovered.

The modified MMPC algorithm is presented in Algorithm 2. The function $Ind(X,T \mid Z)$ return true if x and t are conditionally independent given Z. As for the modified PC algorithm, the Check_Splits function is used to discover the split-ands or split-ors of the workflow graph. The

MaxMinHeuristic function is modified so that if two x variables are equally mutually informative, the one which is closer in time to the selected t is chosen as the variable representing the maximum association.

Algorithm 2 Our Proposed Modified_MMPC Algorithm

```
function MODIFIED_MMPC(t,D)
    Input: target variable $t$, data $D$
    Output: parents and children of $t$
    $G = emptyset$
    repeat
        $<F, assoF> = MAXMINHEURISTIC(t, G)$
        if $assocF <> 0$ then
            $G = G \cup F$
        end if
    until $G$ has not changed
    for all $x \in G$ do
        if $\exists S \subset Gs.t.Ind(x,t \mid S)$ then
            if $Check\_Same\_Info(x,t,s)$ then
                if $time\_difference(x,t) > time\_difference(t,S)$ then
                    $G = G \setminus \{s\}$
                else
                    $G = (G \setminus \{x\}) \cup \{s\}$
                end if
            end if
        end if
    end for
    return G
end function
function MAXMINHEURISTIC(t,G)
    Input: variable $t$, subset of variables $G$
    Output: maximum over all variables of the min association with $t$ relative to $G$, and variable that achieves the maximum
    $assocf = max x \in vMinAssoc(x,t|G) \, if \, !Check\_Splits(x,t)$
    $f = arg max x \in vMinAssoc(x,t|G)$
    return $<f, assocf>$
end function
```

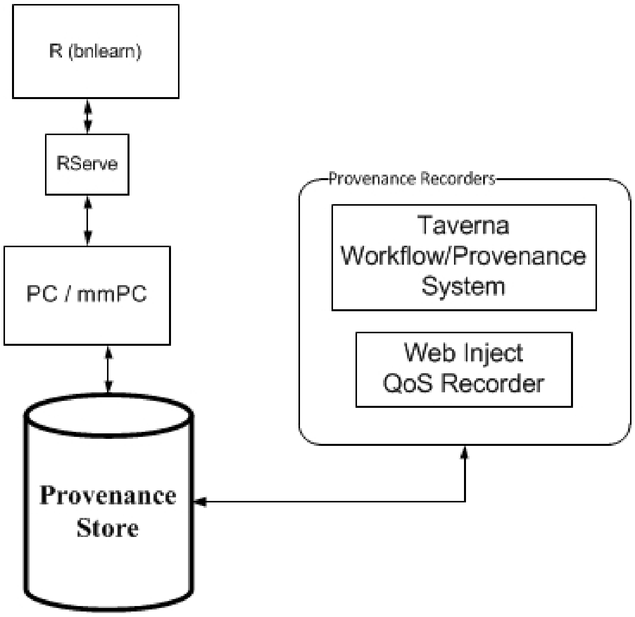


Figure 4.1: Implementation Model.

4.4 Implementation

In order to perform real world and valuable experiments, Taverna (version 2.1) [Tav, 2013] was selected as a practical provenance system to generate real provenance information and was expanded to incorporate the additional features required for our experiments. Taverna does not record timing information such as start or execution time of services during workflow runs. Since Taverna does not record non-functional specifications of web services, Taverna's provenance data model was changed to allow the storage of the QoS values of services. A QoS tracker was added to Taverna to record the QoS specifications of the WSDL services imported by Taverna. The QoS recorder exploits WebInject [Web, 2012], a tool for automated testing of web applications and web services, to monitor the services. We set up the transaction monitors for service-level monitoring of response time and availability of web services. Apart from these parameters, the QoS tracker also keeps track of the execution time and status of execution of services. The conditional independence test from the “bnlearn” library of the R [R, 2012] package was exploited to discover the causal dependencies between services. The “Rserve” [Rse, 2013] server creates the facility to connect to R libraries through our application.

An overview of the implementation model is shown in Figure 4.1. At the beginning, a workflow is created by Taverna. The related services are added to WebInject to record QoS parameters such as timing information. Every time a workflow instance is run, in our implementation model, the provenance data regarding the functional aspects of the workflow are generated through the Taverna system, and the nonfunctional parameters, i.e., QoS values, are produced by the WebInject. All this information is stored in the provenance store and is exploited later by the modified PC and MMPC algorithms for the learning of the workflow structure. These algorithms use R libraries through Rserve to assess the conditional independence tests and discover the causal relationships.

4.5 Case Study and Performance Evaluation

In this section, the experimental results are provided. Three experiments of different scenarios were tested using the proposed approach. Later, the performances of the modified algorithms versus the original ones were assessed.

4.5.1 Three Scenarios

In order to observe the performance of the modified algorithms, 3 different workflow scenarios were considered and tested. The first one consists of a sequential workflow scenario that does not satisfy the faithfulness condition, the second one contains a parallel workflow structure, and the third includes a complex scenario composing of two split constructs.

The web services used by the three scenarios were selected from different service providers available on the internet. the services were were placed together as a workflow in Taverna. Having executed the workflows for multiple times, a dataset of 10,000 rows was created for the experiments.

Case 1

The first scenario is a completely sequential workflow consisting of 4 web services of “WhatisMyIP”, “GetCity”, “WhatsWeather”, and “WeatherImage”. The “WhatisMyIP” service finds the IP of the customer, sends it to the “GetCity” service, which finds the city based on the IP address. The city name is passed to the “WhatsWeather” service and the weather forecast of the city is predicted. Having the forecast, the forecast image is shown to the workflow user presenting the weather condition for his location.

Figure 4.2 shows the structure of the workflow as well as the experimental results. As can be seen from the figure, the original “PC” and “MMPC” algorithms discover some of the edges, while the modified PC and MMPC algorithms discover the complete workflow graph even though the data for this scenario does not satisfy the faithfulness condition. The data stored by the original workflow model represents the graph discovered by Modified algorithms. The relationship between the “GetCity” and “WhatsWeather” is not discovered by the original PC and MMPC algorithms as these two services are conditionally independent given the information provided by the“WhatsMyIP” service. This is due to the similarity between the information that these two services provide. As a result of this, the conditional independence relations in this model can not be represented by a DAG. The modified algorithms extract this relationship.

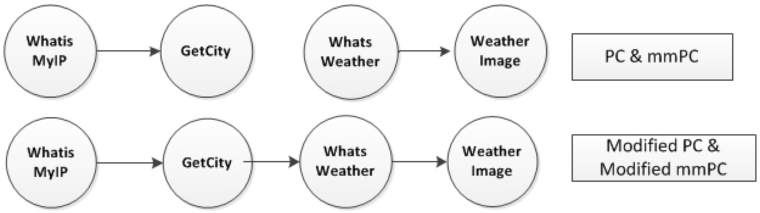


Figure 4.2: Case Study 1: a sequential workflow scenario.

Case 2

The next workflow scenario includes four services, two being run in parallel. Figure 4.3 shows the scenario along with the results of the Bayesian structure learning algorithms. As can be seen from the figure, the original PC and MMPC algorithms can not find the complete parallel structure. This is due to the lack of the faithfulness condition in the structure of the workflow scenario graph. The relation between the ”StockPrice” and ”ProductID” cannot be discovered, as given the ”ProductID” information, these two services are independent. Again, this is due to the similarities between the information these three, services provide. The “ProductID”, “SalePrice”, and “AveragePrice” mutually provide the same information. Similarly, the original PC and MMPC algorithms find the two services of “SalePrice” and “AveragePrice” independent given the “ProductID”. Thus, the conditional independence relations in the data cannot be represented as a DAG.

These issues have been resolved with the modified algorithms by the two added functions. The modified algorithms discover the two services of “SalePrice” and “StockPrice” as parallel, and thus, they are removed from the each other’s conditional sets. Thus, the complete structure is discovered via these modified approaches.

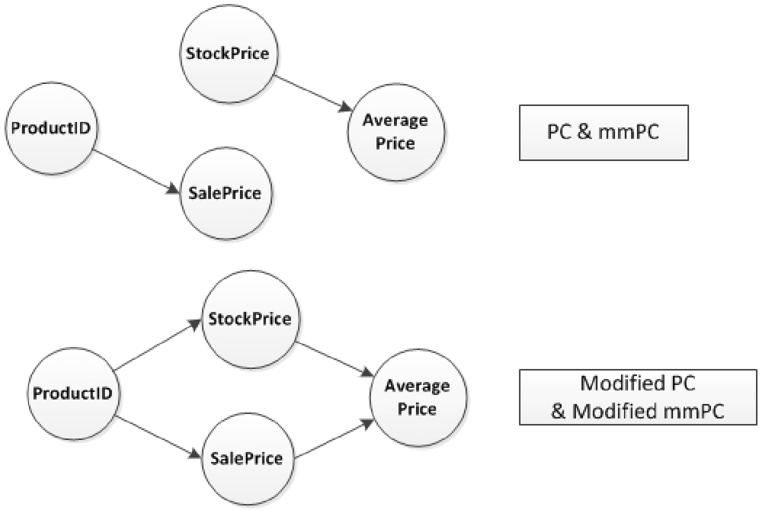


Figure 4.3: Case Study 2: a workflow with parallel parts.

Case 3

As for the third scenario, three separate relevant workflows were considered and the structure learning algorithms were used to discover the graph policy from these workflows. The scenario involves services for receiving and delivering an order in case of a valid credit card payment as well as the availability of the product. Figure 4.4 shows the three paths that can be taken based on the service outputs.

In this scenario, the distinguishable difference between the performances of the modified algorithms versus the original algorithms shows the effects the modifications have had on the extraction of more accurate graphs. Figures 4.5 and 4.6 display the results of the modified algorithms as well as the original algorithms. This scenario includes a split-or, a split-and and a join. The original algorithms can not extract many of the edges due to the large dependencies that exist within the service data.

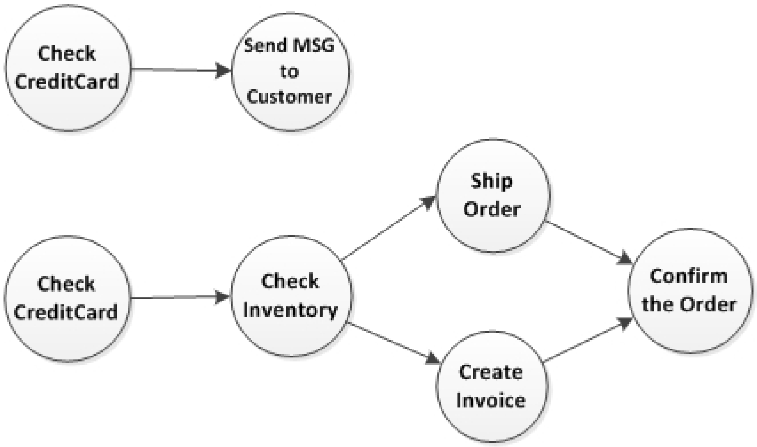


Figure 4.4: Case Study 3: a more complicated scenario with a split-or, a split-and and a join.

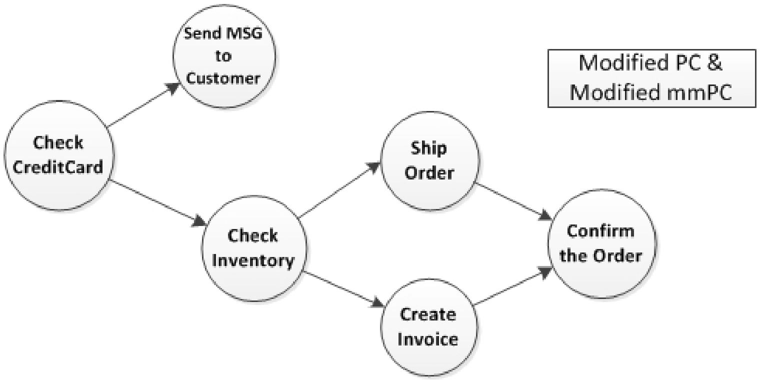


Figure 4.5: Case Study 3: Workflow Results of Modified PC and MMPC

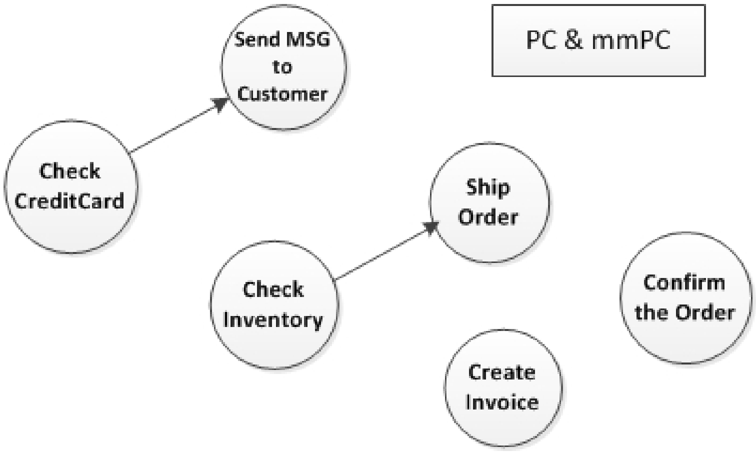


Figure 4.6: Case Study 3: Workflow Results of PC and MMPC

4.5.2 Experimental Results

We evaluated the performance of the modified algorithms with regards to execution time. The original algorithms of PC and MMPC were compared with the modified ones in terms of workflow sizes and execution time. For the experiments, data values of sequential workflows of sizes 5 to 25, with increments of 5, were randomly generated. The experiments were conducted 10 times on different datasets. As can be seen from the graph shown in Figures 4.7 and 4.8, the modified algorithms have a steeper slope, with the PC algorithm having the better performance overall. The modified PC and MMPC algorithms consume more time since the Check_SAME_INFO and Check_Mutual functions increase the number of conditional independence tests the original algorithms use. The MMPC algorithm uses more conditional tests compared to PC, since both the forward and backward phases, perform tests of independence.

4.6 Conclusions and Future Work

In this chapter, we exploited constraint-based Bayesian structure learning algorithms to extract the structure of workflows from provenance data. The output values of services were used to discover the data flow between services along with timing information to provide control flow information. Provenance information was used, since it provides the appropriate amount of data gathered over time, and therefore, makes it suitable for learning. The two algorithms of PC and MMPC were modified in order to better discover the workflow models of the scenarios which do not support the faithfulness condition. PC, MMPC and the modified algorithms were assessed using 3 different scenarios. As the results presented, despite of the fact that some workflow scenarios might not

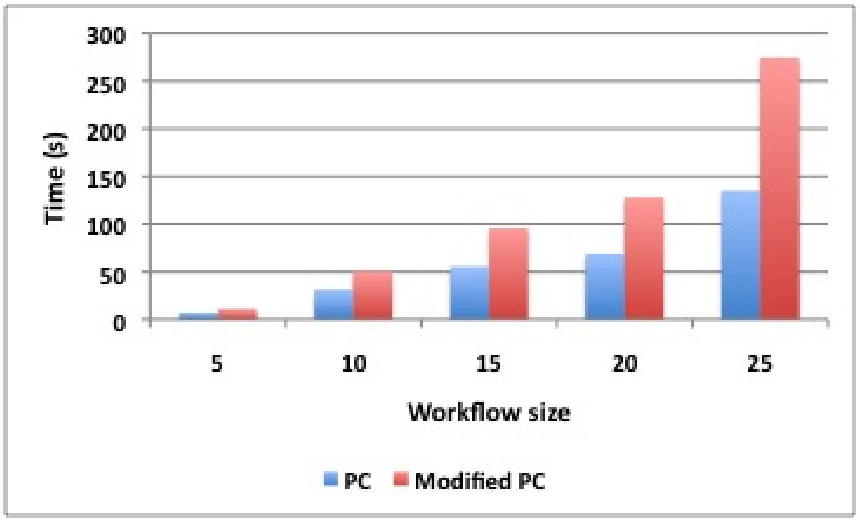


Figure 4.7: Performance Comparison of MMPC and modified MMPC

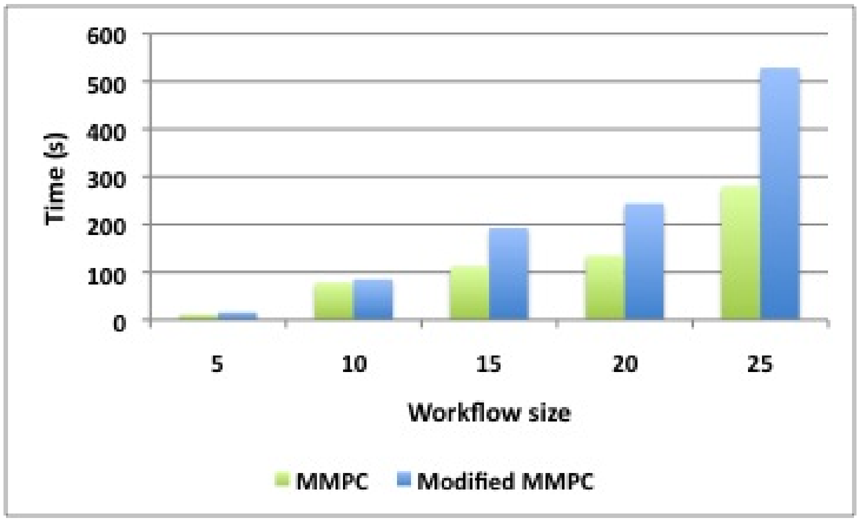


Figure 4.8: Performance Comparison of PC and modified PC

follow the faithfulness condition, the changes applied to both algorithms (PC and MMPC) provide complete and robust structures. The benefit of mining accurate workflow structures comes at the price of higher execution times. Both modified algorithms take longer to find the correct workflow structure.

The current algorithm has some limitations. For the workflow structures with multiple parallel branches where each branch includes several sequential services, the proposed modifications need to be extended. The "CHECK *SAME* INFO" function only considers one conditional variable and thus the structures with multiple services in and-splits or or-splits can not be discovered properly. In addition, assessing the threshold value for time difference calculations needs to be improved for such structures.

Since we only considered services with one output, future work can involve running experiments that include services with multiple output values. Having experimented with the constraint-based Bayesian approaches, score-based algorithms can also be modified and compared against the constraint ones in terms of performance and accuracy.

Chapter 5

Automatic Service Composition Using POMDP and Provenance Data

Service composition is the process of combining services in a specific order to achieve a specific goal, whereby the initial and goal states are determined in advance. The service composition problem is very similar to standard planning problems, since the idea is to discover a path between the initial and goal states. In service composition, the composition of services identifies this path. In this chapter, we exploit provenance information along with Partially Observable Markov Decision Processes (POMDP) to compose services automatically. The POMDP method has been used in the literature for the purpose of robot planning and navigation. In this research, we argue that due to partial observability of service and system states, the POMDP approach provides better solutions for the QoS-aware service composition in dynamic workflow environments. For the purpose of solving the POMDP, service details and the POMDP distributions are learnt from the provenance store. Provenance data contains information regarding workflows, services, their specifications and execution details. This information facilitates the service composition process to be performed more intelligently and efficiently. This chapter has been published in [Naseri and Ludwig, 2013b].

5.1 Introduction

Service composition addresses the problem of automatically placing the services together in a special order to achieve one or more predetermined goal(s). Since one single service is usually not sufficient to fulfill the requirements of a user, thus, a set of appropriate services are selected and composed. The composite service provides more valuable functionalities than a single service, while enhancing the reusability of services as well. A composite service is also referred to as a workflow and includes a set of atomic services together with the control and data flow.

As discussed in [Milanovic and Malek, 2004], a composition method should satisfy several requirements, such as connectivity, non-functional QoS properties, and scalability. The connectivity between the composed services should be reliable. The composition should also address the non-functional QoS properties such as response time, availability, reliability, etc. And, finally, since business transactions can be complex and composed of several services, the composition approach should scale with increasing numbers of composed services.

In general, a service composition approach is performed in two main steps: the first phase, which is referred to as the planning phase, discovers the services that provide the functionalities required by the user. It then generates a set of plans based on the functional parameters of the services. As there might be two or more service implementations for one task, a selection between the execution plans for the service composition is required. The set of functionally equivalent service implementations corresponding to an abstract task, i.e., abstract service, are referred to as concrete services. The non-functional properties of services, QoS values, are used to differentiate between these services. QoS parameters are used to evaluate how well a service composition serves the customer. Generally, these values are presented by the service provider while publishing an advertisement as a service level agreement. The second phase, i.e., the selection phase, calculates the aggregated QoS of the generated plans and selects the best plan that satisfies the non-functional requirements, i.e., QoS specifications of services. The selection of the optimal execution plan that maximizes the QoS values of the composition is an NP-hard problem. Since discovering the optimal execution plan can be time consuming, some simplifications have been assumed for service composition problems [Milanovic and Malek, 2004].

Some service composition approaches relax the QoS constraints to achieve better performance in terms of time. A service composition without constraints can be solved more efficiently in time. However, the optimal execution plan generated by these approaches might exceed the user's budget limit.

The other way of reducing the complexity is to exploit local maximization approaches instead of global ones. The local maximization approaches look for the service implementation with the best QoS for each task instead of evaluating the objective function for that particular QoS property for each execution plan. These methods allow the modeling of the service composition problem as dynamic programming methods [Gao et al., 2006], [Li et al., 2007], or Multi-Constraint Path Problems (MCPP) [McIlraith, 2002].

On the other hand, as there are usually multiple QoS parameters, it is not possible to get the best value for all properties without using multi-objective optimization approaches. Thus, in order to relax the complexity of the service composition problems, single objective optimization approaches are exploited versus the multi-objective ones. The mapping between these problems is done by aggregating the multiple objective functions to a global one in order to use the principles of the single objective optimization.

In this chapter, we exploit the Partially Observable Markov Decision Processes (POMDPs) method [Pom, 2013] along with provenance information for service composition and selection. The POMDP framework has been used for modeling a variety of real-world sequential decision processes. Its application areas mostly include robot navigation problems, machine maintenance, and planning under uncertainty in general. For robot navigation, regardless of the quality or quantity of the sensing hardware deployed on the robot, from its point of view, the robot will have an incomplete view of its environment. With this partial observability, the POMDP model can provide the formal basis for autonomous behaviour in these domains. Machine maintenance involves any machine that requires periodic maintenance due to deterioration of its internal components over time. For this application, POMDPs are used to obtain an inspection/replacement policy that either optimizes the operating costs or the production capacity of the machine [Kaelbling et al., 1998].

The POMDP methods we exploit in this chapter use the basic dynamic programming approaches for solving POMDPs. For all algorithms, the approach solves one stage at a time and works backwards in time. Many of the algorithms use Linear Programming (LPs) to solve POMDPs. In order to assess the POMDP distributions, provenance information are exploited. As mentioned in Chapter 2, in workflow systems, the provenance of a workflow presents information about the workflow process, inputs/outputs of services, intermediate data objects and the QoS specifications of services. Having a large provenance store of previous executions of services and workflows, we plan to perform service composition and selection using the POMDP technique and provenance data.

An expressive language, that supports flexible descriptions of models and data, facilitates reasoning and automatic discovery and composition. Therefore, the service composition approaches mostly exploit the semantic descriptions of services as well as their QoS specifications from service repositories or service providers to perform the composition or selection. The service composition requirements which are entitled in [Gil, 2005], can be satisfied through provenance information as a robust provenance trace provides multiple layered presentation of provenance [et al., 2007]. A layered architecture and engine for automatically generating and managing workflow provenance data is considered in provenance systems which fulfills the requirements of the workflow composition process. These layers include: an abstract description of the workflow, an instance of the abstract model by presenting bindings and instances of the activities, provenance of the execution of the workflow, including specification of services and run-time parameters, and, finally, execution time specific parameters, including information about internal states of the activities. Thus, the provenance information provides a rich source of data for service selection and composition purposes and facilitates automatically composition of services and selection of appropriate services that provide certain QoS requirements.

The remainder of this chapter is organized as follows: In Chapter 5.2, related work is described. In Chapter 5.3, we present how service composition can be modelled as a POMDP and discuss the process taken towards assessing the POMDP distributions using provenance information. In Chapter 5.4, the implementation details of the model along with a case study are presented. In Chapter 5.5, we present the experiments conducted with different numbers of abstract and concrete services using various POMDP algorithms. The last section provides the conclusion of this study.

5.2 Related Work

Based on the simplifications discussed in the previous section and other criteria, service composition has been modeled by several approaches. As described in [Hoffmann et al., 2007], the service composition problem can be viewed as a planning problem. Some of the research work, which exploit planning approaches for service composition include [McDermott, 2002], and [Medjahed et al., 2003b]. Rule-based planning is an approach being used to generate composite services from high level declarative descriptions. The method presented in [Lammermann, 2002] uses composability rules to determine whether two services are composable.

Some approaches such as the ones presented in [Waldinger, 2000b] and [Gil, 2005] exploit theorem-proving for service composition. In [Waldinger, 2000b], the available services and user requirements are described in a first-order language. Then, constructive proofs are generated with certain theorem provers. At the end, service composition descriptions are extracted from particular proofs. [Gil, 2005] uses propositional variables as identifiers for input/output parameters and uses intuitionistic propositional logic for solving the composition problem.

A planning problem can be described as a multi-tuple characterized by the set of all possible states of the world: the initial state of the planner, the set of goal states the planning system should attempt to reach, the set of actions the planner can perform, and the transition relations which specify the semantics of each action describing the state each action results in when executed. The state of a service composition model, which interacts with services, is described by the messages it sends and/or receives. The information contained in each message can be interpreted as the partial description of the current world state. The set of actions the planner can perform is mapped to the set of web service operations. A web service operation is specified by its name, and its input and output message types. Service operations are considered as the actions available to the planning system. Each action in a planning system has preconditions and effects. The preconditions should hold prior to the execution of the action, while the post-conditions should hold after the execution of the action. Service descriptions are used to interpret the states of a service. The input/output messages, sent through service transactions, are used for describing the precondition/effects of service executions. [Hoffmann et al., 2007]

Some background knowledge regarding the semantics of the operations are required for deducing the preconditions and effects of operations from the input and output document schemas. Semantic mark-up languages such as OWL [OWL, 2004] have been used in literature for this purpose.

The work presented in [Doshi et al., 2005] argues that classical planning approaches are not suitable for web service composition as web service invocations are not deterministic. As discussed in this work, a decision-theoretic planning technique such as Markov Decision Processes (MDPs) [Puterman, 1990] better address this issue. As mentioned in [Hoffmann et al., 2007], when describing the state of the world, there is a problem that is not normally encountered in planning systems which is the "partial observability of states". We can only know as much about the current state of the world as is described in the small set of documents. AI planning and Markovian approaches have focused on the situations where the state of the environment is fully observable, instead POMDPs provide a general planning and decision making framework for an agent to act optimally in partially observable domains. It consists of a set of states, a set of actions that the agent can execute, and a set of observations.

The POMDP model augments a well-researched framework of MDPs to situations where an agent cannot reliably identify the underlying environment state. Thus, POMDPs expand the application of MDPs to many realistic problems. It should be mentioned that the generality of POMDPs has the drawback of high computational cost.

Thus, due to partial observability of service and system states, in this chapter, we argue that the POMDP approach is a suitable model for the QoS-aware service composition problem compared to other planning approaches. The POMDP methods can extract composition models that involve structures with non-deterministic branches. Most importantly, as services are unreliable, there are many factors that can affect the status of a service. Thus, exploiting a solution that supports partial observability of the results addresses the issues that would arise in the dynamic service environment. In addition, a POMDP problem is the same as a planning problem, and similarly, given a complete and correct model of the world dynamics and a reward structure, an optimal policy is provided by this method.

5.3 Modeling and Methodology

5.3.1 POMDP Dynamics

The dynamics of the POMDP model are described by the set of states S, actions A, observations O, along with state transition function T, observation function Z, and the reward function R. The state transition function $T : S \times A \longmapsto \triangle(S)$ represents a probability distribution over world states ($\triangle(S)$ denotes the set of all probability distributions over S) for each world state and agent action. $T(s,a,s')$ assesses the probability of ending in state s' given that the agent starts in state s and takes action a.

The reward function $R : S \times A \longmapsto \mathbb{R}$ maps the states and actions into numerical rewards. It represents the expected immediate reward gained by the agent for taking each action in each state. $R(s,s',a)$ represents the expected reward on state transition s to s' given action a. $Z : S \times A \longmapsto \triangle(O)$ is the observation function, which for each action and resulting state provides a probability distribution over possible observations. $Z(s',a,o)$ stands for the probability of making observation o given that the agent took action a and reached state s'.

The goal of the POMDP problem solving task is to select actions as to maximize the reward collection. The optimal behavior in a POMDP requires access to the entire history of the process. As the agent does not know the exact state it is in, it must maintain a probability distribution, known as the belief state, over the possible states. A belief state is a statistic for the history. This means that optimal behavior can be achieved using the belief state in place of the history. A belief state b is simply a probability distribution over the set of states S with $b(s)$ being the probability of occupying state s.

Given a belief state b, in order to compute the resulting belief state b', basic rules from probability theory, Bayes Rule and the independence assumption inherent in the POMDP model, are used. The next belief state depends only upon the previous belief state and the immediate transition taken. Equation 5.1 depicts how transition and observation probability distributions are used toward updating the belief state:

$$b'(s') = \frac{Z(s',a,o)\sum_{s\in S} b(s)T(s,a,s')}{P(o|a,b)} \tag{5.1}$$

Being in a particular belief state b, taking action a, and receiving observation o, the next belief state can be determined. As we are having finite numbers of actions and observations, given a belief state, the number of future belief states are finite.

The agents are desired to act in such a way as to maximize some measure of the long-run reward received. To achieve this, the most straightforward farmework is the infinite-horizon discounted model, in which we sum the rewards over the infinite lifetime of the agent, but discount them geometrically using discount factor $0 < \gamma < 1$. The agent should act so as to optimize the following formula:

$$E[\sum_{t=0}^{\infty} \gamma^t r_t] \tag{5.2}$$

According to this model, rewards which are received earlier in the agent's lifetime have more value to the agent. Although the infinite lifetime is considered, the discount factor ensures that the sum is finite. The larger the discount factor (closer to 1), the more effect future rewards have on current decision making.

5.3.2 Web Service Composition as POMDP

A Web services programming interface is described using WSDL [WSD, 2001], which specifies properties of a web service such as its functionality, location and invocation interface. These interfaces are exploited for the automatic composition of services along with the services' QoS properties, which facilitate dynamic service selection. As mentioned before, we exploit a single objective function for composition purposes. In the context of QoS-aware service composition, there are n QoS properties that have to be optimized. These QoS properties can have conflicts between each other in a way that one, such as availability, should be maximized while another, such as response time, has to be minimized. In order to map multi-objective optimization to single-optimization, the Simple Additive Weighting (SAW) [Strunk, 2010] method is exploited. This method aggregates the objective functions in order to use the principles of the single objective optimization function. For this purpose, QoS properties have to be normalized and summed up to a global QoS value that is then to be maximized.

As mentioned earlier, since the web service environment is dynamic, the agent would not be able to guarantee successful service execution. On the other hand, the state information that we obtain cannot be complete due to the limitation of the document and dynamic nature of the environment. On the other hand, the services are non-deterministic and might be unreliable. As a result, the service invocation outcomes can not be fully known in advance, which make the states not fully observable. Thus, the POMDP model is exploited and in order to model the service composition as a POMDP, the following mappings are required: The status of each task node represents a state, the operations the services perform are mapped to POMDP actions, and accordingly, the invocation results of service operations are mapped to observations of actions. In this work, the service states and invocation results are assumed to be discrete quantities. As for the rewards, the QoS values of the services are to be used to accomplish the service selection. The SAW method is applied to the values of the QoS parameters to obtain a global QoS value.

The dynamics of the POMDP model are to be learnt from the provenance store.We assume that we have a large dataset of the previous executions of different types of workflows. The provenance data required should include information about services, their executions (including input/output parameters and data objects), and QoS parameters such as execution time, response time, cost, status, etc. This information is being used for the POMDP modeling as well as the assessment. The service operations' information provide us with the list of actions, service outputs are used to model the observations and service states are being extracted from the semantic descriptions of services.

To calculate the probability distributions for the POMDP approach, provenance information is exploited along with the ML method. Bayes Rule is applied to each probability, and along with the ML method, the probability values are assessed. This process is performed automatically through the system. First, the list of the service states and actions are extracted from the provenance store. Then, using the list of states and actions, the transition and observation distributions are calculated.

In the following, we describe the information and procedure we use to compute the POMDP distributions. The transition probability for each action, i.e. $T(s,a,s')$ assesses the probability of reaching state s' given that the workflow policy starts in state s and takes action a. In order to assess this probability using provenance information, the ML method is applied to the service states along with timing information. To assess the ML method for the state transition probabilities, for each action we determine the number of state transitions from state i to state j with regard to the total number of transitions available from state i. The transition probability estimation for our model is computed based on the following equation:

$$P(s' \mid s,a) = \frac{n_{ij}}{n_j} \tag{5.3}$$

where for service a, n_{ij} denotes the number of transitions from state i to state j, and n_i denotes the total number of transitions from state i. The start and execution time of services decide about the state orderings. For example, as for service a, the number of times a state transition from state i to state to j has occurred is calculated using the starting and execution time of state i along with the starting time of state j: i.e. for service a, the consequent state transitions from i to j are discovered based on the following:

$$time_{start}(j) = time_{start}(i) + time_{execution}(i) \tag{5.4}$$

The following equation states how this distribution is assessed:

$$T(s,a,s') = P(s'|s,a) = \frac{n^a_{s's}}{n^a_s} \tag{5.5}$$

where $n^a_{s's}$ is the total number of data rows in the provenance data set with current state of s' and the previous state s for action a, and n^a_s is the total number of rows with state s for action a.

Having assessed the probabilities for all the individual states, for each action, the observation matrix is calculated. As the modeling suggests, the observations are assumed to be discrete.

Similarly, as for the transition matrix, the observation matrix values are assessed through the ML method and the status of the execution of services. The $Z(s', a, o)$ is determined by computing the probability as follows:

$$P(o \mid s', a) = \frac{o_{ij}}{o_j} \tag{5.6}$$

where o_{ij} denotes the number of data rows in the provenance dataset where being in state s' and taking service a, the observation o was recorded. The o_j presents the total number of data rows where, having been in state s', action a was taken.

$Reward(s, s', a)$ is defined as the response time, cost, and or any aggregated QoS parameters associated with service a during a state transition. The goal of the service selection phase of the composition problem is to find the solution, which optimizes the aggregated QoS value. The aggregated value of QoS parameters of all data rows associated with service a is averaged and stored as the reward/cost for that service in the rewards matrix.

Having modeled the service composition as a POMDP, the composition is formed by solving the model that generates the optimal service composition policy graph. POMDP models can be solved using exact solution techniques.

An exact solution to a POMDP yields the optimal action for each possible belief over the world states. The optimal action optimizes the expected reward of the agent over a possibly infinite horizon. The sequence of optimal actions is known as the optimal policy of the agent for interacting with its environment. The exact method calculates the optimal policy by generating two arrays of V, for the value, and Ω, for the policy. At the end of the algorithm, Ω contains the solution, and V contains the discounted sum of the rewards to be earned on average by following that solution from state s. As any POMDP can be reduced to a continuous belief-state MDP, the value iteration phase for POMDPs is the same as continuous MDPs. It is a standard method for finding the optimal infinite horizon policy using a sequence of optimal value functions.

The Value iteration algorithm will iteratively generate a set of vectors, V, which will be evolved using the previous stage vectors. Each vector in the next stage, V', is constructed from the immediate rewards and the transformation of V using the POMDP functions. A vector in V has a particular strategy associated with it. Each vector at a stage represents the value of acting according to the particular current and future strategy for that vector. Selecting a vector at a stage is the same as selecting a particular course of action at a stage and a particular future action strategy.

As for value iteration, it is important to be able to extract a policy from a value function. For policy iteration, it is important to be able to represent a policy so that its value function can be calculated easily. The policy iteration phase represents and improves the policy. The methods applied on this phase usually consist of two steps of policy evaluation and policy improvement. The policy evaluation phase discovers a policy tree by finding the action associated with each node n and the successor node of n after receiving observation o. The policy improvement step performs a standard dynamic programming backup during which the value function is transformed into an improved value function [Braziunas, 2003].

5.3.3 POMDP Algorithms

Several POMDP algorithms exist that are distinguished by the way the value iteration is performed. The enumeration algorithm [Cassandra et al., 1994] is an exact POMDP algorithm and conceptually the simplest of all the exact algorithms. It first generates all possible vectors by ignoring the belief state and later on uses Linear Programming to discard useless vectors. In order to construct a vector, an action and a vector in V for each observation should be selected. Thus, large numbers of vectors can be generated, of which many are not useful, since they are dominated by other vectors over the entire belief space. These vectors can be eliminated at the expense of some computing time, but, regardless, enumerating over the vectors takes a long time even for some small problems.

The witness algorithm [Monahan, 1982] tries to find the best value function for each of the actions separately. Unlike some algorithms, it does not consider all the actions all the time. As described, we can represent V and V' using collections of policy trees, respectively. This algorithm, first finds a collection of policy trees that represent the expected reward by taking action a from belief state b. It then defines regions for a vector and looks for a point where that vector is not dominant. Once these functions are discovered, they are combined into the final V' value function. Simply put, the witness algorithm is using linear programming to find a single point called “witness” with the fact that $V' \neq V$. If a witness is found, it is used to determine a new vector by solving a linear program and this process continues.

The incremental pruning algorithm [Zhang and Liu, 1996] combines elements of the enumeration and the witness algorithms. Similar to the witness algorithm, it considers constructing sets of vectors for each action individually and then focuses on each one observation at a time. The incremental Pruning algorithm can solve the problems that cannot be solved within a reasonable time in the Witness algorithm. It breaks down the value function V' as a combination of simpler value functions [Pom, 2013].

5.4 Implementation and Case Study

5.4.1 Implementation

Our implementation of the presented model exploits the POMDP solver presented in [POM, 2013] to discover the policy graph and to perform the service composition. The solver has implementations of the enumeration algorithm, the witness algorithm, the incremental pruning algorithm, and few more. The code uses linear programming to solve POMDPs. The POMDP solver receives an input file of the POMDP problem in a certain format and solves it using the selected POMDP algorithm, discount factor, and other settings. The discount factor is a value between 0 and 1 which is used to make the total reward finite. It is used during the value iteration algorithm. This factor dictates the relative usefulness of future rewards compared to immediate rewards. Based on the value of this factor, the rewards received later get discounted, and contribute less than the current rewards.

In order to model the service composition as a POMDP and to save the model in the appropriate file format, we implemented a Java program which extracts the services, states, and observations from the provenance data. We exploited the Taverna provenance system [Tav, 2013] to generate provenance information, but since Taverna does not support QoS recording, Taverna's provenance information was augmented with state variables and QoS values which were measured and generated using other measuring tools, such as WebInject [Web, 2012]. Our program then assesses the

POMDP probabilities from the provenance store, creates the transition and observation matrices using the proposed model, normalizes each matrix row, and calculates the rewards. These values are then formatted into the POMDP solver input file, which is then solved by the solver. The enumeration algorithm, the witness algorithm, and the incremental pruning algorithm are the three POMDP approaches exploited for the purpose of evaluation.

5.4.2 Case Study

To better present how the service composition can be modeled and solved through a POMDP, a case study is provided which addresses the following scenario:

A manufacturer wants to deliver an order to a retailer. The manufacturer might satisfy the order in one of several ways. He first checks for the availability of the order in his inventory. If the order is available in his stock, he will then assemble the order and ship it to the retailer's address. In case the manufacturer is out of stock, he checks his supplier for availability. The last option would be to check the market stock availability for the order.

Based on the described case study, the services for this scenario include abstract services for checking the inventory, checking the supplier, checking the stock market, assembling the order, and shipping the good. The results of these services would be either *yes* or *no* entries. In case of the availability of the stock, or successful assembly and shipment, the result would be a *yes* entry, and in the other cases a *no* entry. The QoS values associated with each service include the service cost and execution time.

The following model suggests the list of POMDP actions, observations, and states that are modeled for this scenario. According to our described model, the services are mapped into POMDP actions. Since each action can result in a success or failure, the observations for each action include

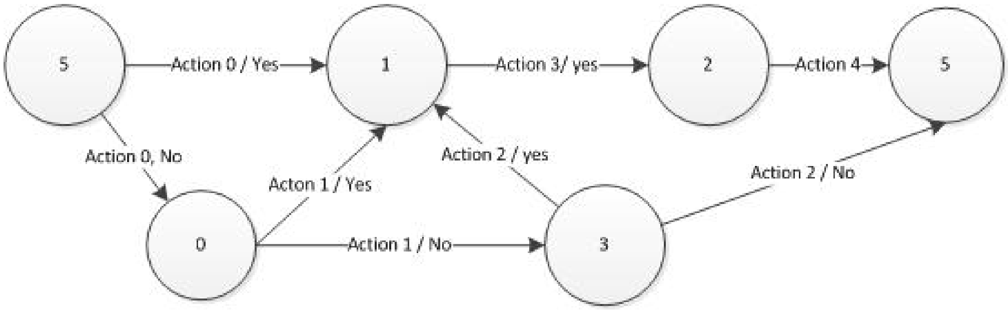


Figure 5.1: Case study scenario

a {YES,NO} set. As for the states, the initial state starts with checking the availability of the stock in the inventory. Each action based on its success or failure would result in a new state that is described according to the actions. An *end* state is considered as a dummy state.

The following are the states, observations, and actions for this case study.

States: *Invent_Avail* (inventory is available); *Supp_Avail* (supplier is available); *Market_Avail* (market is available); *Shipped_Order* (order is shipped); *Assemble_Order* (order is assembled); *End* (final state).

Observations: *Inv_avl_YES* (inventory is available); *Inv_avl_NO* (inventory is not available); *Sup_avl_YES* (supplier is available); *Sup_avl_NO* (supplier is not available); *Mar_avl_YES* (market is available); *Mar_avl_NO* (market is not available); *Assmbl_YES* (good is assembled); *Assmbl_NO* (good is not assembled); *Ship_YES* (good is shipped); *Ship_NO* (good is not shipped).

Actions: *Check_Inventory_Availability* (check availability of inventory); *Check_Supplier_Availability* (check availability of supplier); *Check_Market_Availability* (check availability of market); *Assmble_Order* (assemble order); *Ship_Order* (ship order).

Figure 5.1 displays the scenario.

5.5 Experiments and Results

A provenance store of previous executions of the workflow paths using different concrete services and the POMDP parameters and input file were generated. The QoS of response time was the only QoS parameter used in this experiment. The experiments were done on an Intel Pentium 4 CPU 2.4 GHz machine with 1 GB of RAM.

First, the three POMDP algorithms were verified with the case study shown in the previous section. Then, in order to evaluate the scalability of the proposed service composition approach, a set of experiments were performed scaling different numbers of abstract and concrete services.

5.5.1 Verification of Method

To perform the verification experiment, for each abstract service, 5 concrete services providing the same functionality but different QoS values were considered. The discount factor was set to 90%.

The results presented in Figure 5.2 are the POMDP policy graph generated by the solver, which depict the service composition with optimal services. The figure shows the structure of the service composition found by the generated POMDP policy. As can be seen, the services have been composed correctly and the exact model structure is discovered since Figures 5.1 and 5.2 are identical.

As the workflow model in the case study presents, the POMDP approach is able to discover complex structures with parallel or-splits, a split at which just one branch is active at a time. The POMDP can also extract the parallel and-splits, a split at which all branches are active at a time. Since POMDP treats and-splits the same way as or-splits, they are discovered similarly.

As for the selection phase, the POMDP approach selects the optimal path by choosing the

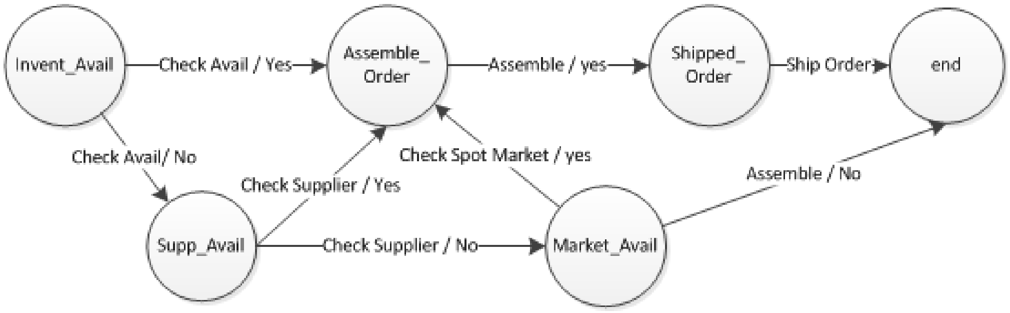


Figure 5.2: Policy graph generated by the POMDP solver

concrete services that provide the least cost and response time for each abstract service. The total execution time for solving this scenario was 1.5 seconds.

5.5.2 Scalability Analysis

Typically, organizations have many processes consisting of different numbers of activities in the form of services. The number of activities will be very different depending on the process application area. In this section, we present the scalability results of the algorithm along with the experiments done using the different POMDP algorithms. Since POMDP is solved using linear programming methods, the scalability of our approach was assessed by three sets of experiments.

The first experiment was performed with a constant number of abstract services and variable numbers of concrete services. The same scenario presented in the case study with 6 abstract services was used for this experiment. The number of concrete services was set to 5 at the beginning and was incremented by 5 consecutively up to 25 services. The discount factor was set to 80%. The POMDP algorithm selected for this experiment was the incremental pruning algorithm. Figure 5.3 shows the results. To assess the scalability of the approach with regards to the number of concrete services, the total number of services involved in each experiment is determined. For the first mea-

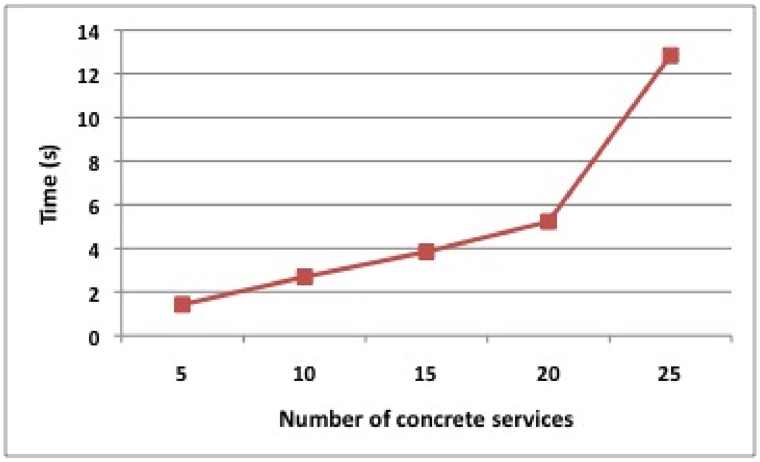


Figure 5.3: Performance evaluation for different numbers of concrete services with a constant number of abstract services

surement point, 6 abstract services times 5 concrete services results in 30 services in total. As for the last measurement point, where we have 6 abstract services and 25 concrete services for each abstract service, the number of services in total is 120. Therefore, for 5 concrete services a total of 30 services, and for 25 concrete services a total of 120 services are involved when searching for the appropriate policy graph. These experiments suggest that the POMDP approach is capable of providing scalability with regards to realistic numbers of concrete services.

For the second experiment, we enlarged the size of the service composition problem by changing the number of abstract services incrementally by 5, while keeping the number of concrete services to 5. The POMDP algorithm selected for this experiment was the incremental pruning algorithm. The experimental results are displayed in Figure 5.4. For a workflow size of 5 abstract services, each having 5 concrete services, the execution time is 1.8 seconds, whereas for 25 abstract services the execution time is 27.8 seconds.

The last experiment assesses the performance of the three POMDP algorithms (enumeration, incremental pruning, and witness) on the service composition. The number of abstract services

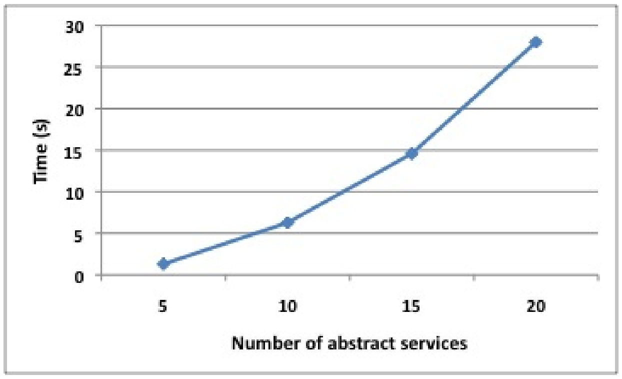


Figure 5.4: Performance evaluation for different numbers of abstract services with a constant number of concrete services

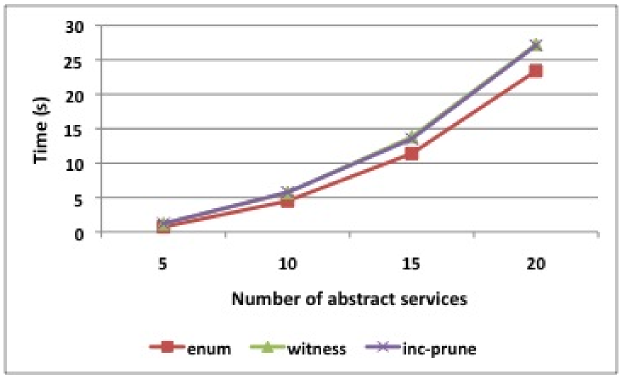


Figure 5.5: Performance evaluation with regard to various POMDP algorithms

are increased by 5 services and the execution time is evaluated. The graph in Figure 5.5 displays the results. It can be observed that all three algorithms show a similar trend, with the enumeration algorithm performing slightly better compared to the others for the experiments.

should comment *on what ranges of services would commonly be encountered in practice *how things would be different if had much greater degrees of uncertainty

5.6 Conclusion

This Chapter showed an approach to service composition and selection by exploiting the provenance information along with partially observable Markov decision processes to compose the services of a workflow automatically. Provenance data contains information regarding workflows, services, their specifications and execution details. We modeled the QoS-aware service composition as a POMDP, learning the service details from the provenance store. In particular, we presented how the service composition problem can be modeled as a POMDP. We argued that since service composition can be seen as a planning problem, due to the dynamic environment of services and the uncertainty, POMDP is an appropriate approach for service composition. It is important to mention that as a Web service environment is a dynamic environment, provenance data might not have the information of all the possible states for a certain service. On the other hand, the services are dynamic as well and they might change through time. This information might not be available in the provenance store. As a result, the observability of states are partial and the agent cannot reliably identify the underlying environment state.

Experiments were performed to assess the scalability of the model, and the performance of several POMDP algorithms was evaluated. The method showed reasonable scalability and the algorithms provide similar performance in terms of execution time. The proposed approach is also applicable to QoS-aware composition cases where the optimal selection of services is not desired but instead a range of required QoS values are specified. This requires a small modification on the reward assessment.

As the degrees of uncertainty in a structure increases, the search for discovering the policy graph becomes more difficult. As a result, the execution time of the POMDP methods would increase. A future experiment would include assessing the performance of the model with regards to increasing degrees of uncertainty for a workflow structure. Applying hierarchical POMDPs to the composition problem, which is likely to result in better performance, is another research exoeriment which will be followed in the future work. Since hierarchical POMDPs require an abstract hierarchy of actions, they provide a suitable approach to improve the scalability of the proposed method.

CHAPTER 6

CONCLUSIONS AND FUTURE DIRECTIONS

6.1 Conclusions

In this research, we discussed the concept of provenance, provenance application areas and some of the architectures and systems which provide facilities for presenting, recording, and querying this information. The literature review reveals that not much research has been devoted to the applications of provenance data. We discussed that a large provenance store of previous executions of services and workflows provides an appropriate environment for reasoning and learning. Thus, having discussed the motivations and requirements, we proposed an architecture that addresses the current challenges with workflow and services using provenance information. The architecture is composed of components for workflow and service trust or performance evaluations, automatic selection and compostion of services, workflow mining and pattern discovery, as well as a workflow structure refinement. The connection and interactions between components suggest that each component can provide lower level functionalities individually or a more complex functionality in combination with other components. We demonstrated four of the main components of the architecture and proposed a novel approach to address the issues with regards to each component. The list of major contributions made in this thesis are the following:

- Exploiting provenance information to address the current challenges and problems in service oriented environments. These challenges include service composition and selection, workflow mining, workflow evaluation, and refinement.

- Proposing a multifunction architecture which applies statistical learning methods along with provenance information to solve the services and workflow problems.
- Presenting a new approach for evaluating trust of services and workflows which is based on HMMs. The trend of trust of the workflow was assessed and the effect of the stationary assumption in the HMM model was investigated.
- Mining workflow structures using Bayesian structure learning approaches and provenance information. The main contribution is the exposition of a new approach that discovers the causal relationships between the services using the data flow as well as control flow information stored in provenance information. The proposed method can discover workflows with sequential and parallel structures. In addition, it is capable of discovering the service graph composed of workflows and services that are related and belong to the same subject or area.
- Exploiting the POMDP solutions towards automatic composition and selection of services using the previous history of workflow runs. The proposed method takes the partial observability of services environments into account for service composition and selection.

6.2 Future Directions

The research provided in this thesis suggested an architecture for targeting the current issues and problems with web services and workflows based on provenance information. While studies were run and new approaches were followed for most of the components of the architecture, however, some problems and issues still remain open. The proposed architecture can be augmented with other services to provide more functionality, robustness, and reliability. To increase the stability and intelligence of the architecture, each component can provide feedback to the provenance store to feed the provenance data with the information learned through the process the component follows to achieve its goal. The recorded data can be exploited to train the system dynamically through time. As a result of this process, the components will operate in a more intelligent and robust manner.

As for the components individually, more research can be done to provide each component with more functionality. The workflow evaluation component can be expanded to give information on performance of each service individually over time, as well as the changes occurring in services' QoS trends. In order to compose services more efficiently, the information provided by the evaluation component could be used to compose more robust workflows. In addition, the component's functionality can be improved in a way that it generates various service compositions, if possible, along with the probability values for overall workflow QoS values. As for the workflow structure learning component, the approach presented can be developed to also identify implicit novel relationships referred to as hidden connections between services.The research in this thesis did not focus on the workflow refinement component. This component could also be added to a future version of the architecture and can cooperate with both the structure learning and evaluation components.

Further on, the usability of new approaches for each component could be investigated. Time series and other statistical methods could be studied to be exploited for the purpose of workflow evaluation.

As for the experiments, the Taverna workflow system was used to create workflows and provenance data automatically. The new architecture could provide services on top of the Taverna system so that the components could be used along with the Taverna workflow System. Experiments and scenarios could be designed to assess the performance of the architecture and investigate how efficiently the components cooperate.

BIBLIOGRAPHY

Web service description language (wsdl) version 1.1. http://www.w3.org/TR/wsdl, 2001. [Online; last accessed 2010].

The triana project. http://www.trianacode.org/collaborations/, 2003. [Online; last accessed 2012].

Web ontology language (owl). http://www.w3.org/TR/owl-features/, 2004. [Online; last accessed 2010].

Resource description framework (rdf). http://www.w3.org/RDF/, 2004. [Online; last accessed 2010].

Sparql query language for rdf. http://www.w3.org/TR/rdf-sparql-query/, 2004. [Online; last accessed 2010].

Preserv 0.3.1. http://twiki.pasoa.ecs.soton.ac.uk/bin/view/PASOA/SoftWare, 2005. [Online; last accessed 2013].

Jess rule engine. http://herzberg.ca.sandia.gov/, 2008. [Online; last accessed 2010].

Workflow, from wikipedia. http://en.wikipedia.org/wiki/Workflow, 2009. [Online; last accessed 2012].

Kepler reporting 2.4 suite. https://kepler-project.org/users/whats-new/reporting-2.4-suitereleased, 2011. [Online; last accessed 2013].

Probability distribution, from wikipedia. http://en.wikipedia.org/wiki/Probability_distribution, 2011. [Online; last accessed 2013].

The r project for statistical computing. http://www.r-project.org/, 2012. [Online; last accessed 2012].

Webinject, web application and web services test tool. http://webinject.org/, 2012. URL http://webinject.org/. [Online; last accessed 2012].

Kepler project. https://kepler-project.org/, 2013. [Online; last accessed 2010].

Maximum liklihood(ml) method. http://en.wikipedia.org/wiki/Maximum_likelihood, 2013. [Online; last accessed 2012].

Mysql database software. www.mysql.com, 2013. [Online; last accessed 2012].

Pomdp solver. `http://www.cs.brown.edu/research/ai/pomdp/`, 2013. [Online; last accessed 2010].

Pomdp tutorial. `http://www.cs.brown.edu/research/ai/pomdp/tutorial/index.html`, January 2013. [Online; last accessed 2012].

Rserve. `http://cran.rproject.org/web/packages/Rserve/index.html`, 2013. [Online; last accessed 2012].

Taverna workflow system. `http://www.taverna.org.uk/`, 2013. [Online; last accessed 2012].

Extensible markup language (xml). `http://www.w3.org/TR/owl-features/`, 2013. [Online; last accessed 2011].

W. M. Aalst and B. F. Dongen. Discovering workflow performance models from timed logs. In *In Proceedings of the First international Conference on Engineering and Deployment of Cooperative information Systems*, 2002.

J. Abellan, M. Goomez-Olmedo, and S. Moral. Some variations on the pc algorithm. In *Proceedings of the third European workshop on probabilistic graphical models (PGM'06)*, pages 1–8, 2006.

R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large database. *in Proceedings of the ACM SIGMOD International Conference on Management of Data*, 22(2):207–216, 1993.

R. Agrawal, D. Gunopulos, and F. Leymann. Mining process models from workflow logs. In *Proceedings of the 6th International Conference on Extending Database Technology: Advances in Database Technology*, Heidelberg, 1998. Springer Verlag.

Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB '94, pages 487–499, San Francisco, CA, USA, 1994. Morgan Kaufmann Publishers Inc.

R. Aiello. *Workflow Performance Evaluation*. PhD thesis, University of Salerno, Italy, 2004.

I. Altintas. Lifecycle of scientific workflows and their provenance: A usage perspective. In *Proceedings of the 2008 IEEE Congress on Service*. IEEE, 2008.

D. Ardagna and B. Pernici. Global and local qos guarantee in web service selection. In *Proceedings of Business Process Management Workshops*, pages 32–46. Proceedings of Business Process Management Workshops, 2005.

J. H. Kim B. Sin. Nonstationary hidden markov model. *Signal Processing*, 45(1):31–46, 2008.

R. Berbner, M. Spahn, N. Repp, O. Heckmann, and R. Steinmetz. In proceedings of int'l conf. on web services. In *Heuristics for QoS-aware Web Service Composition*, 2006.

S. Bongkee and H. K. Jin. Nonstationary hidden markov model. *Signal Processing*, 46:31–46, 1995.

S. Boreman. The expectation maximization algorithm, a short tutorial. [Online; last accessed 2012], 2009.

U. Braun. Incremental workflow mining for process flexibility. *Business Process Modeling, Development, and Support (BPMDS)*, 6, 2006.

D. Braziunas. Pomdp solution methods. Technical report, Technical Report, 2003.

P. Buneman and W. C. Tan. Provenance in databases, why, how, and where. *Foundations and Trends in Databases*, 1:379–474, 2007.

A. R. Cassandra, L. P. Kaelbling, and M. L. Littman. Acting optimally in partially observable stochastic domains. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, 1994.

A. Chebotko, L. Cui, F. Xubo, L. Zhaoqiang, L. Shiyong, H. Jing, and F. Fotouhi. View: a visual scientificworkflow management system. In *Proceedings of IEEE Congress on Services*. IEEE, 2007.

S. M. Cruz, P. M. Barros, and et al. Provenance services for distributed workflows. In *Eighth IEEE International Symposium on Cluster Computing and the Grid*. IEEE, 2008.

Y. Cui. Tracing the lineage of view data in a warehousing environment. *ACM Transactions on Database Systems (TODS)*, 25(2):179–227, 2000.

Y. Cui and J. Widom. Practical lineage tracing in data warehouses. In *Proceedings of the ICDE*, pages 367–378, 2000.

R. Daly, Q. Shen, and S. Aitken. Learning Bayesian networks: approaches and issues. *The Knowledge Engineering Review*, 26:99–157, 2001.

M. Day. Provenance and data-intensive science. Working Draft, 2005.

A. K. Alves deMedeiros, B. F. van Dongen, W. M. P. van der Aalst, and A. J. M. M. Weijters. Process mining for ubiquitous mobile systems: an overview and a concrete algorithm. In *Ubiquitous Mobile Information and Collaboration Systems (UMICS 2004)*. Springer Verlag, HeidelbergSpringer Verlag, Heidelberg, 2004.

B. F. Dongen and W. M. P. van der Aalst. A meta model for process mining data. In *Proceedings of the CAiSE'05 Workshops*, pages 309–20, 2005.

P. Doshi, R. Goodwin, R. Akkiraju, and K. Verma. Dynamic workflow composition using Markov decision processes. *International Journal of Web Services Research*, 2:1–17, 2005.

J. Kim et al. Provenance trails in the wings-pegasus system. *Concurrency and Computation: Practice and Experience*, 20, 2007.

M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery: An overview. In *Advances in Knowledge Discovery and Data Mining*, pages 1–34. AAAI Press / The MIT Press, Menlo Park, CAAAAI Press / The MIT Press, Menlo Park, CA, 1996.

S. Fine, Y. Singer, and N. Tishby. The hierarchical hidden Markov model: Analysis and applications. *Machine Learning*, 32:41–62, 1989.

G. Forman. An extensive empirical study of feature selection metrics for text classification. *J. Mach Learn. Res.*, 3:1289–1305, 2003.

G. Fox and D. Gannon. Workflow in grid systems. *Concurrency and Computation: Practice and Experience*, 18, 2006.

J. Freire, D. Koop, E. Santos, and C. T. Silva. Provenance for computational tasks: A survey. *Computing in Science & Engineering*, 10(3):11–21, 2008.

N. Friedman, I. Nachman, and D. Pe'er. Learning Bayesian network structure from massive datasets: the sparse candidate algorithm. In *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence (UAI-99)*, 1999.

N. Fuhr and T. Rolleke. A probabilistic relational algebra for integration of information retrieval and database systems. *Journal of ACM Transactions on Information Systems*, 15:32–66, 1997.

W. Gaaloul and C. Godart. Mining workflow recovery from event based logs. *Business Process Management*, pages 169–185, 2005.

W. Gaaloul, S. Alaoui, K. Baina, and C. Godart. Mining workflow patterns through event-data analysis. In *In Proceedings of the 2005 Symposium on Applications and the internet Workshops*, pages 226–229. IEEE Computer Society, 2005.

Y. Gao, J. Na, B. Zhang, L. Yang, and Q. Gong. Optimal web services selection using dynamic programming. In *Proceedings of the 11th IEEE Symposium on Computers and Communications (ISCC '06)*, Washington, DCIEEE Computer Society, Washington, DC, 2006. IEEE Computer Society.

Y. Gil. Workflow composition: Semantic representations for flexible automation. *Workflows for E-Science: Scientific Workflows for Grids, Book Chapter*, pages 244–257, 2005.

J. Golbeck. Filmtrust: Movie recommendations using trust in web-based social networks. In *the Proceedings of Consumer Communication and Networking Conference*, 2006a.

J. Golbeck. Combining provenance with trust in social networks for semantic web content filtering. In *Proceedings of International Provenance and Annotation Workshop*. IPAW 2006, 2006b.

P. Grawth, J. Sheng, M. Simon, M. Steve, T. Victor, T. Sofia, and M. Luc. An architecture for provenance systems: executive summary. Technical report, University of Southampton, 2006. Electronics and Computer Science.

M. Greenwood and et al. Provenance of e-science experiments, experiments from bioinformatics. *in the UK OST e-Science second All Hands Meeting, East Midlands Conference Centre, Nottingham*, pages 223–226, 2003.

P. Groth, Sh. Jiang, S. Miles, S. Munroe, V. Tan, S. Tsasakou, and L. Moreau. An architecture of provenance systems. Technical report, 2005.

V. Guralnik and J. Srivastava. Event detection from time series data. In *Proceedings of the Fifth ACM SIGKDD international Conference on Knowledge Discovery and Data Mining (San Diego*, pages 33–42, California, 1999. ACM.

J. D. Hamilton. *Time Series Analysis*. Princeton, NJ ISBN 0-691-04289-6, princeton u. press edition, 1994.

J. Herbst and D. Karagiannis. Integrating machine learning and workflow management to support acquisition and adaptation of workflow models. *International Journal of Intelligent Systems in Accounting, Finance and Management*, 9:67–92, 2000.

J. Herbst and D. Karagiannis. Workflow mining with inwolve. *Computers in Industry*, 53:245–64, 2004.

J. Hoffmann, P. Bertoli, and M. Pistore. Web service composition as planning, revisited: in between background theories and initial state uncertainty. In *Proceedings of the 22nd national conference on Artificial intelligence*, pages 1013–1018. AAAI Press, 2007.

K. Huang and C. Chang. Efficient mining of frequent episodes from complex sequences. *Inf. Syst*, 33(1):96–114, 2008.

S. Hwang and W. Yang. On the discovery of process models from their instances. *Decision Support System*, 1(34):41–57, 2002.

F. Jelinek. Self-organized language modeling for speech recognition. Technical report, IBM T.J. Watson Research Center, 1985.

X. JingHui, L. BingQuan, and W. XiaLong. Principles of non-stationary hidden markov model and its applications to sequence labeling task. In *Proceedings of the Second International Joint Conference on Natural Language Processing*, 2005.

L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence 101*, pages 99–134, 1998.

S. Karlin and H. M. Taylor. *A First Course in Stochastic Processes, Second Edition*. 1975.

A. Krogh, S. I. Mian, and D. Haussler. A hidden Markov model that finds genes in e.coli dna. *Nucleic Acids Research*, 22:4768–4778, 1994.

S. Lammermann. *Runtime Service Composition via Logic-Based Program Synthesis*. PhD thesis, Department of Microelectronics and Information Technology, Royal Institute of Technology, June 2002.

M. Last and A. Kandel. Data mining in time series databases. *series in machine perception Artificial Intelligence*, 57, 2004.

J. Ledlie and et al. Provenance-aware sensor data storage. In *21st International Conference on Data*. Engineering Workshops, 2006.

Y. Li, J. Huai, T. Deng, H. Sun, H. Guo, and Z. Du. Qos-aware service composition in service overlay networks. In *Proceedings of IEEE International Conference on Web Services*, pages 703–710. IEEE, July 2007.

D. Margaritis. *Learning Bayesian Network Model Structure from Data*. PhD thesis, School of Computer Science, Carnegie-Mellon University, Pittsburgh, 2003.

D. McDermott. Estimated-regression planning for interactions with web services. In *Proceedings of the 6th International Conference on AI Planning and Scheduling*, France, AAAI PressToulouse, France, AAAI Press, 2002. Toulouse.

S. McIlraith. Adapting golog for composition of semantic web services. In *Proceedings of the 8th International Conference on Knowledge Representation and Reasoning (KR2002)*, France-Toulouse, France, April 2002. Toulouse.

B. Medjahed, A. Bouguettaya, and A. K. Elmagarmid. Composing web services on the semantic web. *The VLDB Journal*, 12(4), 2003a.

B. Medjahed, A. Bouguettaya, and A. K. Elmagarmid. Composing web services on the semantic web. *The VLDB Journal*, 12:333–351, 2003b.

N. Milanovic and M. Malek. Current solutions for web service composition. *IEEE Internet Computing*, 8(6):51–59, December 2004.

Simon Miles, Sylvia C. Wong, Weijian Fang, Paul Groth, Klaus peter Zauner, and Luc Moreau. Provenance-based validation of e-science experiments. In *In ISWC*, pages 801–815. Springer-Verlag, 2005.

G. E. Monahan. A survey of partially observable Markov decision processes: theory, models, and algorithms. *Management Science*, 28, 1982.

L. Moreau and et al. Concurrency and computation: Practice & experience. *The First Provenance Challenge*, 20(5):409–418, 2008.

L. Moreau, O. F. Rana, and D. Walker. Case for support: Pasoa provenance aware service oriented architectures. In *last retrieved*. 2013.

T. Murata. Petri nets: Properties, analysis and applications. In *Proceedings of the IEEE 77.4*, pages 541–580, 1989.

K. P. Murphy. A survey of pomdp solution techniques: Theory, models, and algorithms. *Management Science*, 28, 1982.

S. Narayanan and S. McIlraith. Simulation, verification and automated composition of web service. In *Proceedings of the 11th International World Wide Web Conference*, Hawaii, USAHonolulu, Hawaii, USA, 2002. Honolulu.

M. Naseri and S. A. Ludwig. A multi-functional architecture addressing workflow and service challenges using provenance data. In *Proceedings of Workshop for PhD. Students in Information and Knowledge Management (PIKM) in conjunction with the 19th ACM Conference on Information and Knowledge Management (CIKM)*, Toronto, Canada, 2010.

M. Naseri and S. A. Ludwig. Evaluating workflow trust using hidden markov modeling and provenance data. *Data Provenance and Data Management in eScience*, 426, 2012.

M. Naseri and S. A. Ludwig. Extracting workflow structures through bayesian learning and provenance data. In *Proceedings of the 13th International Conference on Intelligent Systems Design and Applications*, Malaysia, 2013a.

M. Naseri and S. A. Ludwig. Automatic service composition using pomdp and provenance data. In *Proceedings of the 2013 IEEE Symposium Series on Computational Intelligence (SSCI)*, Singapore, 2013b.

E. Newcomer and G. Lomow. *Understanding SOA with Web Services*. Addison Wesley, 2005.

J. Cheney P. Buneman and S. Vansummeren. On the expressiveness of implicit provenance in query and update languages. *ACM Trans. Database Syst.*, 33(4):28:1–28:47, 2008.

N. Prat and S. Mandick. Measuring data believability: a provenance approach. In *Proceedings of the 41st Annual Hawaii International Conference on System Sciences*, 2008.

M. L. Puterman. Markov decision processes. *Handbooks in Operations Research and Management Science*, 2:331–434, 1990.

L. Rabiner and B. Juang. An introduction to hidden Markov models. *ASSP Magazine, IEEE*, 3.1: 4–16, 1986.

L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, pages 257–286, 1989.

S. Rajbhandari, I. Wootten, A. S. Ali, and O. F. Rana. Evaluating provenance-based trust for scientific workflows. In *presented at the Proceedings of the Sixth IEEE International Symposium on Cluster Computing and the Grid*. IEEE, 2006.

S. Rajbhandari, O. F. Rana, and I. Wootten. A fuzzy model for calculating workflow trust using provenance data. In *Proceedings of 15th ACM Mardi Gras conference: From lightweight mashups to lambda grids: Understanding the spectrum of distributed computing requirements tools, infrastructures, interoperability, and the incremental adoption of key capabilities*. ACM, 2008.

M. J. Sannella. *Constraint Satisfaction and Debugging for Interactive User Interfaces*. PhD thesis, University of Washington, 1994.

G. Schimm. Mining exact models of concurrent workflows. *Computers in Industry*, 53:65–81, 2004.

Y. L. Simmhan, B. Plale, and D. Gannon. A survey of data provenance techniques. Technical report, Indiana University, 2005.

P. Spirtes, C. N. Glymour, and R. Scheines. *Causation, Prediction, and Search*. MIT Press, Cambridge, MA, No. of pages 543, 1 edition, 2000.

L. Srinivasan and J. Treadwell. *An Overview of Service-oriented Architecture, Web Services and Grid Computing*. HP Software Global Business Unit, 2005.

A. Strunk. Qos-aware service composition: A survey. In *Proceedings of the IEEE 8th European Conference on WebServices (ECOWS*. IEEE, 2010.

M. Szomszor and L. Moreau. Recording and reasoning over provenance data in web and grid services. In *Int Conf. on Ontologies Databases and Applications of SemanticsConf. on Ontologies, Databases and Applications of Semantics*, 2003.

W. Tan and M. Zhou. *Business and Scientific Workflows: A Web Service-Oriented Approach (IEEE Press Series on Systems Science and Engineering)*. WILEY, 2013.

A. Tiwari, C. J. Turner, and B. Majeed. A review of business process mining: State-of-the-art and future trends. *Business Process Management Journal*, 14:5–22, 2008.

H. Truong and T. Fahringer. Online performance monitoring and analysis of grid scientific workflows. *Advances in Grid Computing - EGC*, 3470:1154–1164, 2005.

I. Tsamardinos and L. E. Brown. The max-min hill-climbing Bayesian network structure learning algorithm. *Journal of Machine Learning*, 65:31–78, 2006.

W. van der Aalst, T. Weijters, and L. Maruster. Workflow mining: discovering process models from event logs. *IEEE Transactions on Knowledge and Data Engineering*, 16(9):1128–1142, September 2004.

W. M. P. van der Aalst and A. K. Alves de Medeiros. Process mining and security: Detecting process executions and checking process conformance. *Electronic Notes in Theoretical Computer Science*, 121:3–21, 2005.

W. M. P. van der Aalst and A. J. M. M. Weijters. Process mining: A research agenda. *Computers in Industry*, 53(3):231–244, April 2004.

W. M. P. van der Aalst, A. J. M. M. Weijters, and L. Maruster. Workflow mining: which processes can be rediscovered? Beta Working Paper Series, WP 75, Eindhoven University of Technology, Eindhoven, 2002.

W. M. P. van der Aalst, A. K. Alves de Medeiros, and A. J. M. M. Weijters. Genetic process mining. In G. Ciardo, editor, *Applications and Theory of Petri Nets*. Springer Verlag, Heidelberg, 2005.

R. Waldinger. Web agents cooperating deductively. In *Proceedings of FAABS*, Greenbelt, MD, USA, 2000a.

R. J. Waldinger. Web agents cooperating deductively. In *Proceedings of the First International Workshop on Formal Approaches to Agent-Based Systems-Revised Papers (FAABS '00)*, 2000b.

Y. R. Wang and S. E. Madnick. A polygen model for heterogeneous database systems: The source tagging perspective. *Very Large Data Bases (VLDB)*, 1990.

J. Widom. Trio: A system for integrated management of data, accuracy and lineage. In *Proceedings of the 32nd international conference on Very large data bases*, pages 1151–1154, 2006.

A. Woodruff and M. Stonebraker. Supporting fine-grained data lineage in a database visualization environment. In *Proceedings of the Thirteenth International Conference on Data Engineering*, pages 91–102, 1997.

D. Wu, E. Sirin, J. Hendler, D. Nau, and B. Parsia. Automatic web services composition using shop2. In *Workshop on Planning for Web Services*, ItalyTrento, Italy, 2003.

N. L. Zhang and W. Liu. Planning in stochastic domains: Problem characteristics and approximation. Technical report, HKUST-CS96-31, Dept. of Computer Science, Hong Kong University of Science and Technology, 1996.

J. Zhao. Linked data and provenance in biological data webs. *Semantic Web for Health Care and Life Sciences: A Review of the State of the Art*, 10(2):139–152, 2009.