The Dissertation Committee for Yuqun Zhang
certifies that this is the approved version of the following dissertation:

Modeling and Predicting Data for Business Intelligence

Committee:

Dewayne E. Perry, Supervisor

Suzanne Barber

Sarfraz Khurshid

Milos Gligoric

Guowei Yang

Modeling and Predicting Data for Business Intelligence

by

Yuqun Zhang, B.E.; M.S.

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

December 2016

Dedicated to my parents Minzhan Zhang and Qianyi Mao.

Acknowledgments

I wish to thank all the people who helped me, accompanied me, and supported me throughout my PhD studies.

I would like to dedicate my deepest thanks and love to my PhD advisor Professor Dewayne E. Perry. I have the tremendous honor to work under his supervision and learn from him academia- and personality-wide. Professor Perry provides me the precious opportunity to complete my PhD studies in UT and guides me to the path of being an independent researcher. He gives me boundless freedom for discovering problems and pursing solutions of my own interests and unconditionally supports me whenever I feel unconfident or hesitant to proceed. He sets an inspiring example of life altitude towards pure joys of life and contempt of utilitarianism. I could not make this far and become what I am without him, and I expect to be a better me in the future by carrying on everything I have learned from him.

I want to thank Professor Don Batory, Professor Suzanne Barber, Professor Sarfraz Khurshid, Professor Milos Gligoric, and Professor Guowei Yang, who serve in my qualifying and dissertation committee and have been helpful to make this thesis happen. Their insights to my research and suggestions on improving the thesis enlightens not only my PhD work but also my upcoming career.

I want to thank Professor Christine Julien, Dr. Meiru Che, and Dr. Herb Krasner for their generosity of offering helping hands at different stages of my PhD studies. These kindness carried me through every difficulty and challenge that occurred to me before I could finally obtain my degree.

I want to thank my previous advisor Professor Wendi Heinzelman from the University of Rochester. She lead me to the path and showed me the charm of research and encouraged me to pursue my PhD program. She keeps giving me advices for my PhD studies and future careers and has no hesitation of recommending me for job applications.

I am thankful to the amazing staff of the Department of Electrical and Computer Engineering Melanie Gulick, RoseAnna Goeway, and Cayetana Garcia.

Even for a person who sets the strongest minds to chase dreams, he would be weakened when he feels lonely. Luckily, I have been loved and supported by so many dearest friends who are respectable with their innocent souls. My gratitude goes to the following people who I have established wonderful friendship with In UT: Lingming Zhang, Chao Wang, Jinru (Lisa) Hua, Rui Qiu, Xi (James) Zheng, Tianyi Zhang, Ripon Saha, Yen-Jung Chang, Wei-Lun Hung, Mengshi Zhang, Lin Huang, Yumin Li, Junyuan Ding, Tony Petz, Seth Holloway and Jung Ryu. I would also like to extend my appreciation to Yuhong Yao, Chen-Hsiang Feng, Xiuyin He, Zhe Gao, and Guanghui Zhou from the University of Rochester and Bao Li, Zhou Lu, and all the other friends who have been supportive for my PhD studies.

Lastly, I would like to send my love to my family. I have been physically parted from my parents for my PhD studies, during when they have been the source of my power for always. I never felt they are thousands of miles away from me. And I am ready to shorter the distance in any dimension in the future.

Modeling and Predicting Data for Business Intelligence

Publication No. ____________________

Yuqun Zhang, Ph.D.
The University of Texas at Austin, 2016

Supervisor: Dewayne E. Perry

Business intelligence is an area where data and actionable information can be analyzed and provided to make more informed business actions. In general, any technique that contributes to better business decisions can be categorized in business intelligence techniques. Particularly, business process management is a subarea that focuses on improving business performance by managing and optimizing business processes; management data mining is a subarea that applies data mining techniques to gain better business performances.

In business process management, a business process is a collection of relevant, structured activities or tasks that produce specific services or products for certain business goals. Business process modeling refers to the activities of representing intra or inter-organization processes, such that the current processes can be analyzed or improved. While abundant business process modeling techniques and their associated analyses have been proposed to capture different aspects of business processes, modern business processes can be very complicated such that many properties, such as performance optimization and evaluation, still cannot be accurately described and understood.

Management data mining refers to applying data mining techniques in multiple domains of business managements, e.g., supply chain management, marketing analysis. Typical research topics include building models to provide feedbacks for skewing supply chain policies or marketing strategies. Traditional research tend to build generic models given specific scenarios, that are argued to easily cause inaccuracies with more granular disturbances.

My thesis focuses on approaches handling the challenges in business process optimization and evaluation and its associated data analysis. Specifically, I propose a data-centric technique for modeling composite business activities by including components of data, human actors, and atomic activities. Furthermore, I explore this technique in two major dimensions. First, by applying this technique in workflow-based business processes, I explore the possibility of reconstructing these processes by modifying the execution order of business activities, and develop efficient algorithms to approach optimal temporal performance for data-centric business processes. Second, I build a symbolic process generator to stochastically generate symbolic data-centric business processes that can be used to analyze their properties and evaluate optimization approaches according to end-users' specification. Moreover, I zoom in a granular data type of inventory management process and build data mining models to forecast it.

The major contributions of my thesis include: *1)* proposing a data-centric business process modeling technique that emphasizes business artifacts compared with traditional workflow-based modeling techniques; *2)* developing approaches to optimize the temporal performance of the data-centric business processes; *3)* applying my symbolic process generator so that data-centric business processes can be simulated and provided with quick and inexpensive analyses. *4)* building data mining models for forecasting inventory shipments based on real-world data sets.

Table of Contents

Acknowledgments **v**

Abstract **viii**

List of Tables **xv**

List of Figures **xvi**

Chapter 1. Introduction **1**

1.1 Research Context 2

1.2 My Solutions 4

1.2.1 Data-Centric Business Process Modeling (SOSE'14) . . 4

1.2.2 Temporal Performance Optimization upon Data-Centric Business Processes (SCC'14, IJSC) 5

1.2.3 Generating Symbolic Business Processes for Data-Centric Business Process Evaluation (SCC'15) 6

1.2.4 Predicting Inventory Shipments (SCC'16) 8

1.3 Contributions 10

1.4 Organization 11

Chapter 2. Background **13**

2.1 Business Process Modeling Techniques 13

2.2 Business Process Optimization 16

2.3 Evaluation of Business Process Optimization 17

2.4 Predictive Modeling for Business Management 20

Chapter 3. Data-Centric Business Process Modeling **22**

3.1 Summary 22

3.2 Approach 23

3.2.1 initial_data 24
3.2.2 input_data 24
3.2.3 global_data 25
3.2.4 consumed_data 26
3.2.5 output_data 26
3.2.6 final_data 26
3.2.7 role 27
3.2.8 atomic_activity 27
3.2.9 subprocess and cluster 28
3.3 An Example 30

Chapter 4. Temporal Performance Optimization upon Data-Centric Business Processes 33

4.1 Summary 33
4.2 Approach 34
4.2.1 Eligibility for Business Activity Relocation 34
4.2.1.1 Motivation 34
4.2.1.2 An Analogous Representation 35
4.2.1.3 My Solutions 38
4.3 Algorithms of Temporal Optimization 47
4.3.1 Temporal Optimization for A Single Subprocess 48
4.3.2 Temporal Optimization for One Cluster 52

Chapter 5. Generating Symbolic Business Processes for Data-Centric Business Process Evaluation 62

5.1 Summary 62
5.2 Approach 63
5.2.1 Hierarchy 63
5.2.1.1 Activity 64
5.2.1.2 Subprocess 64
5.2.1.3 Cluster 64
5.2.1.4 Process 66
5.2.2 System I/O 66

5.2.3 Symbolization 68
5.2.4 Flow Patterns 69
5.2.5 Easy Access 71
5.3 A Case Study of the Usability of G-DCBP 71
5.3.1 Eligibility Verification for Business Activity Relocation . 71
5.3.2 Algorithms to Improve Temporal Performances 72
5.3.2.1 Time Assignments 72
5.3.2.2 Temporal Gain Calculations and Rankings . . . 73
5.3.3 Implementation of Business Activity Relocation 74
5.3.4 Evaluations 75
5.3.4.1 Scenarios 75
5.3.4.2 Metrics 76
5.3.4.3 Results 77
5.3.4.4 Analysis of Evaluation Results 85

Chapter 6. Predict Inventory Shipments **88**
6.1 Summary 88
6.2 Extracting Inventory Transactions 88
6.3 Predicting Inventory Shipments 90
6.3.1 The Methodology with ARMA 90
6.3.1.1 ARMA Algorithm 90
6.3.1.2 Data Preprocess 92
6.3.1.3 ARMA Evaluation 94
6.3.2 The Methodology with KNN Algorithm 96
6.3.2.1 Data Preprocess 96
6.3.2.2 Primitive KNN Algorithm 98
6.3.2.3 Primitive KNN Evaluation 99
6.3.2.4 Time-Efficient KNN 100
6.3.3 The Methodology with Approximate Nearest Neighbor Algorithm 102

Chapter 7. Conclusion **112**

Appendices **115**

Appendix A. An Example of Data-Centric Business Process **116**

Bibliography **120**

Vita **145**

List of Tables

2.1 Paper survey results of aspects of evaluation for business process optimization 18
2.2 Paper survey results of evaluation w/o comparison for business process optimization 19

4.1 Sets of data types 44

6.1 Subinventory lists 89
6.2 RMSE under different p and q settings for ARMA implementations 95
6.3 RMSE under Primitive KNN implementations 100
6.4 RMSE under Approximate Nearest Neighbor implementations 108
6.5 Average RMSE under different methods 108
6.6 Response time (seconds) for Time-Efficient KNN with validation sets and Approximate Nearest Neighbor implementations 109
6.7 RMSE and the corresponding D of the Time-Efficient KNN with validation sets 110

List of Figures

3.1 Examples of join_cluster, split_cluster, and join_and_split_cluster, that are represented by black dots. 29

3.2 The original BPMN model of purchasing apartments 31

4.1 The BPMN model of purchasing apartments updated by modifying the execution of insurance 36

4.2 The PnF representation of "apartment purchasing" process . . 37

4.3 The PnF representation of the updated "apartment purchasing" process 38

4.4 The two possible updated "apartment purchasing" processes . 39

4.5 An example of a cluster with two subprocesses and the activities eligible to be relocated to them 50

4.6 The updated cluster with activity C2 relocated to the subprocess B and updated eligible activities accordingly 53

4.7 The final updated cluster with the waiting time of subprocess B minimized 54

4.8 Updated working time of subprocess C after C2 being relocated 55

4.9 Updated working time of subprocess b after C2 being relocated 57

4.10 The tables of eligibly-relocated subprocess and the corresponding working time gain maintained by eligible activities 58

4.11 The process to sort the working time gains of all activities and select the corresponding eligibly-relocated subprocesses 59

4.12 The updates of the rest activities and their respective working time gains with respect to the relocated activity and its associated subprocesses 60

5.1 The G-DCBP class Diagram 65

5.2 temporal improvement ratio vs. mean activity working time in light-loaded scenario 77

5.3 occurrence ratio vs. mean activity working time in light-loaded scenario 78

5.4 temporal improvement ratio vs. *mean_number_of_data_in_subprocess* in light-loaded scenario 79

5.5 occurrence ratio vs. *mean_number_of_data_in_subprocess* in light-loaded scenario 80

5.6 temporal improvement ratio vs. mean activity working time in heavy-loaded scenario 81

5.7 occurrence ratio vs. mean activity working time in heavy-loaded scenario 82

5.8 temporal improvement ratio vs. *mean_number_of_data_in_subprocess* in heavy-loaded scenario 83

5.9 occurrence ratio vs. *mean_number_of_data_in_subprocess* in heavy-loaded scenario 84

5.10 temporal improvement ratio vs. *mean_number_of_data_in_subprocess* in heavy-loaded scenario of 50% *mean_data_dependency_rate* . 85

5.11 occurrence ratio vs. *mean_number_of_data_in_subprocess* in heavy-loaded scenario of 50% *mean_data_dependency_rate* 86

6.1 Sample of the inventory transaction data set 91

6.2 Annual shipments of 92000-20 94

6.3 Group past inventory shipments that differ by 24 hours as input data and pair them with the future inventory shipments as output data 97

6.4 Placing non-zero elements of an input inventory shipment to bins with the same dimensions 103

6.5 The evaluation results showing the computation time of the Approximate Nearest Neighbor, Primitive and Time-Efficient KNN implementations when D = 20 and 30 105

6.6 The evaluation results showing the computation time of the Approximate Nearest Neighbor, Primitive and Time-Efficient KNN implementations when D = 40 and 50 106

Chapter 1

Introduction

Business intelligence refers to techniques that analyze and provide data and actionable information such that better business decisions can be made. Some typical business intelligence areas include business process management and management data mining.

In business process management, a business process is an assembly of tasks to accomplish a business goal(s). Recently, the emergence of IT innovations have been applying the concept of workflow systems to assist the execution of business processes and enabling automation in design, operation, and management of business processes [1]. Research about business processes are mainly focused on modeling techniques (e.g., [2–10]), and verifying properties of business processes and workflow (e.g., [11–19]), etc.

Management data mining uses data mining techniques to different domains of business management such that the feedback or the improvements of the current strategies and policies can be provided. For instance, some classic research domains are about building models for marketing policies and customer relations(e.g., [20–26]), and inventory managements (e.g., [27–36]).

1.1 Research Context

Traditional modeling techniques for business processes mainly include flowcharting [2–4], IDEF family [5,6], Petri Nets [7], etc. While each of these graphics-based techniques models business processes with their preferred emphasis, it is argued in [37–39] that these approaches are incomplete, implicit and inadequate in handling the business process modeling for complex business goals. Data-centric business process modeling techniques [11,40,41], on the other hand, emphasize business artifacts, including data relevant to business entities, along with information about the macro-level lifecycle that these entities move through. As opposed to other modeling techniques that only focus how workflows are operated, data-centric business processes enable additional understandings of business artifacts.

In the realm of business process modeling, optimization refers to the automated improvement of business processes using pre-specified quantitative measures of performance (objectives) [42]. A group of researchers believe that ideally, a holistic approach for business process modeling should optimize business processes and eventually generate improved ones [42,43]. Among all the business process metrics, time is an important one. A large portion of the relevant research is based on scheduling, that mathematically models the optimal resource allocation of business processes in terms of specified objectives [44–47]. While these approaches aim to optimize time-relevant business applications, it is argued in [48] that they can only be applied in simplified and ideal business processes where their constraints and objectives can be mathematically modeled.

Another challenge of business process optimization is the difficulty in finding suitable business processes to be studied [49]. On the other hand, given suitable business processes, it can be complicated and time-consuming to capture, verify, and analyze them [50–52]. Although some approaches [49–51] can improve the efficiency of modeling business processes under certain constraints, they still assume the business processes to be used by their studies already exist. Even if they do exist, they may be difficult to be captured and modeled.

In enterprise-level business process managements, Inventory management has been a long-lasting research topic in business process management. Particularly, it can connect other components of business processes, such as manufacture, procurements, and sales. Nowadays, it has become more challenging since the business operations are becoming more intrigue. To handle these challenges, the concept of lean inventory management [27], is introduced such that the efficiency and profit of inventory management can be improved.

Many approaches have been proposed to contribute to lean inventory management [28] [29] [30] [31] [32] [33] [34] [35] [36] [53] [54]. One of them is predicting inventory metrics for helping business actors in making plans and decisions [55] [56] [57] [58] [59]. For instance, forecasting demands that are intermittent and low volume in enterprises with large-scale inventory systems [57], planning safety stock by forecasting incoming demands [56], etc. Most of the these approaches build their models by generalizing inventory management scenarios and evaluate them with historical data or simulations.

1.2 My Solutions

My thesis aims at developing modeling techniques to provide solutions for business process optimization and evaluation and its data analysis. I develop a data-centric business process modeling technique. Furthermore, I develop techniques to optimize the temporal performance of the data-centric business processes by reconstructing them. I also build a symbolic process generator to stochastically generate symbolic data-centric business processes that can be used to analyze their properties and, hence, evaluate my temporal optimization techniques. Lastly, I present a framework to forecast typical business process data, i.e., inventory shipments and have it evaluated with real-world data sets extracted from Oracle EBS systems.

1.2.1 Data-Centric Business Process Modeling (SOSE'14)

My data-centric business process modeling technique explicitly models composite business activities that represent business processes with a well-defined syntax, that is reified from modeling techniques for software processes [60,61]. A composite business activity is specified as a tuple of components: data, human actor, and atomic activity, where data is further modeled to indicate its states and life cycle in workflows within a composite business activity. As in [62], an atomic activity is defined as the activity which executes a single unit of work. That means partial data or activity generated within an atomic activity that does not impact on the business artifacts is not represented. Human actor refers to role executing or involved with atomic activities in the composite activity. Note that the modeling of a composite business activity is subjected to ensuring that all atomic activities are executed by specified roles. For the sake of brevity, I do not take time or location into consideration.

Moreover, a subprocess is defined to be composed of business activities in execution order. Subprocesses that are bounded by the identical convergence or divergence relations are defined as a cluster. A business process is configured by clusters and their placements (*SOSE'14* [63]).

1.2.2 Temporal Performance Optimization upon Data-Centric Business Processes (SCC'14, IJSC)

Assuming deterministic components of business activities that cannot be redesigned in my approach at this point, my time-optimization approach aims at reconstructing business processes by modifying the execution order of business activities, and by relocating business activities from their original subprocesses to others.

With an apriori knowledge (i.e., either known or estimated) of time of each business activity as in [1, 64], my approach is initiated by determining whether a business activity is eligible for relocation in a business process. Given a subprocess that a business activity is not originally located in, the business activity is determined to be eligible for relocation to the subprocess only when it is, as defined in my approach, forwardly moveable, backwardly moveable, or loosely-parallelled moveable to the subprocess.

For any subprocess, given the working time that is defined similar to the total lapse time to complete a process unit (e.g., business activity, subprocess, and process), as in [65, 66], its waiting time is defined as the time duration of its idle state when other subprocesses are being executed. In my approach, the temporal optimization of a subprocess is essentially to reduce its waiting time by filling its idle states, in other words, executing eligible business activities from other subprocesses. By selecting the proper eligible activities according to their temporal gains for reducing the waiting time of target subprocess, my algorithm effectively minimizes its waiting time.

Furthermore, I develop an algorithm to approach optimal average temporal performance of clusters. Similar to the mechanism of selecting eligible activities for a single subprocess, my algorithm to approach optimal average temporal performance of clusters modifies the criteria of temporal gains of eligible activities to emphasize their contributions to reduce the working time of the clusters that it is relocated from/to. (*SCC'14* [67], *IJSC*).

1.2.3 Generating Symbolic Business Processes for Data-Centric Business Process Evaluation (SCC'15)

In this thesis, I introduce an approach, namely G-DCBP, to generate symbolic business processes, rather than discovering real-world business processes, for evaluating approaches to optimize data-centric business processes. While the latter is more desirable, the difficulties in gaining access to them can be significant.

Implemented in Java, G-DCBP generates process components (i.e., data, activities, subprocesses, and clusters) and creates a hierarchy that incorporates them into data-centric business processes. The inputs of G-DCBP are simply the number (stochastic and deterministic) of the process components, where the stochastic inputs are used to derive their response time values by applying pre-determined stochastic models.

Data symbolization is a fundamental factor in demonstrating my symbolic data-centric business processes. In G-DCBP, *data*[1] are symbolized by combining English letters for the different data types (e.g., input_data, output_data) defined in [63]. Accordingly, *activities* are delineated by their associated *data* and data types. *Subprocesses* are represented with serialized *activities* in their execution order and *clusters* are represented as combined parallel *subprocesses.*

Data flow patterns, are considered as another important factor in constructing data-centric business processes and stating the global business effects, properties, and objectives. In G-DCBP, data flows are described by transitions and transmissions of *data* in both macro- and micro-levels. A micro-level data flow illustrates that newly-generated *data*, along with the *data* that is inherited from the executed *activities*, are assigned to the associated unexecuted *activities.* Macro-level data flows are delineated by aggregating micro-level

[1]In this thesis, italic expressions are used to represent components of G-DCBP, while the corresponding regular expressions are used to represent the theoretical concepts

data flows between *subprocesses* and *clusters*.

To validate the efficacy of G-DCBP, its generated symbolic processes is used to evaluate the approach to improve temporal performance of data-centric business processes by changing the execution order of business activities, as proposed in [67], where the two critical theories (eligibility verification for business activity relocation and the algorithm to improve temporal performance) are mapped from the concepts to G-DCBP implementations. The evaluation results reflect the degree of temporal improvements of business processes in different scenarios (*SCC'15* [68]).

1.2.4 Predicting Inventory Shipments (SCC'16)

In this thesis, I present approaches to predict shipments using historical data from Oracle EBS system [69]. Shipment is an important metric in inventory management, that directly reflects enterprises' operating performances. Predicting shipment can bring multi-aspect benefits, such as better planning of slotting, feedback to make manufacture orders, etc.

Oracle E-Business Suite (EBS) system is widely used in enterprises for operation management. Specifically, the studied data is extracted from the inventory management module of the Oracle EBS system in a GPS-manufacturing company, namely A, where inventory transactions are explicitly recorded in multiple dimensions. Typically for any customer, its Oracle EBS system is customized based on a unified standardized functionality. Therefore, to analyze the data from the system, it is necessary to understand the features of the system design. In this thesis, our first task is to recognize the features of the inventory management functionality of A's Oracle EBS system and extract the data that need to be analyzed accordingly.

One way to predict inventory is to directly apply time series forecasting algorithms, e.g., ARMA [70]. Some approaches apply other types of learning algorithms and adjust them to be suitable for time series analysis, such as the K Nearest Neighbors (KNN) algorithm in [71] [72], where for each data point as an output, its input is defined as a group of vectors that occur within every 24 hours. K of those data points whose inputs are nearest to the future data points are averaged to be its predicted value. Both techniques are applied here to predict the inventory shipments. The results suggest that the latter has a better accuracy.

Note that in KNN implementations, the optimal prediction accuracy can be achieved when K equals 1, that is, the predicted data is determined to be the one whose input is the nearest to the input of the predicted data. We then modify the KNN algorithm such that the nearest data point of the predicted data can be quickly identified. Furthermore, to reduce the redundancy of computations, we develop a new algorithm that selects and aggregates the data points, and determines the nearest one to the predicted data from them. The evaluation results indicate that the majority of the test cases can approach similar prediction accuracy with much less computation time.

1.3 Contributions

The contributions in this thesis are listed as follows.

- **Data-Centric Business Process Modeling.** I propose a data-centric business process modeling technique that is based on its prototype concepts, for in-depth understandings of the overall effects of business entities as well as workflow-based business process optimization and evaluation.

- **Temporal Performance Optimization for Data-Centric Business Processes.** I identify under which circumstances the temporal performance of data-centric business processes can be optimized, and develop approaches to automatically optimize it accordingly.

- **Generating Symbolic Business Processes for Data-Centric Business Process Evaluation.** I propose a symbolic business process generator (G-DCBP) to stochastically generate symbolic data-centric business processes for a quick and inexpensive evaluation for approaches of business process modeling and optimization.

- **Simulation Studies.** I conducted simulations for my approach to optimize the temporal performance of data-centric business processes by using the symbolic business processes produced by G-DCBP. The simulation results can validate the efficacy of both the data-centric business process modeling technique and its temporal optimization approaches, and discover the degrees of impacts on the temporal performance improvement from different business process properties.

- **Predicting Inventory Shipments.** I provide rationale to extract and preprocess the studied data such that they can be desirably presented in Oracle EBS systems. Moreover, I design algorithms that tailer the manner of featuring the current data points with past ones and achieve much better response time performance compared and similar prediction accuracy with existing algorithms.

1.4 Organization

The rest of this document is organized as follows.

Chapter 2 presents the background and existing work for the areas that I mainly work on, i.e., business process modeling techniques, optimization, and evaluation approaches.

Chapter 3 presents the data-centric business process modeling techniques.

Chapter 4 presents the approaches to determine the eligibility of business process reconstruction and to optimize temporal performance of data-centric business processes due to different business goals.

Chapter 5 presents the symbolic data-centric business process generator and uses it to evaluate the efficacy of my approaches to optimize the temporal performance of data-centric business processes.

Chapter 6 presents the framework to predict inventory shipments from the real data sets extracted from the Oracle EBS systems.

Chapter 7 concludes the dissertation.

Chapter 2

Background

In this chapter I present the background and some related work in the research areas of business process modeling technique, business process optimization, evaluation of business process optimization, and predictive modeling for business managements.

2.1 Business Process Modeling Techniques

A number of approaches have been used to provide techniques for modeling business processes. Flowcharting, showing the scope of the whole business process and tracking the flows of its associated information, was initially proposed by Schriber in [3]. Subsequent work, such as [4], improves this technique by introducing diagrams to include the principles of decision sciences. BPMN [2] standardizes the flowcharting diagrams to bridge the communication gap between business process design and implementation, and is now widely used [15, 16]. The IDEF family integrates business processes and data structures. IDEF0 [5] is designed to model the activities in terms of organizations, while IDEF3 [6] is modeled as process flows to indicate how organizations work. Role activity diagrams [73] concentrate on modeling roles with their associated activities and relations in the context of roles execution as critical resources. These graphics-based techniques excel in the global demonstrations of business processes. However, graphics can be simplified when business processes become too complex or expanded to provide full details. The XML-based modeling languages, such as BPEL [8] and BPML [62], are designed as the executable languages for specifying the activities of business processes to support web services. Though capable of "expressing executable processes that address all aspects of enterprise business processes", it is argued that they are user-unfriendly and fail to convey the concepts of human actors and data models [37].

Business activities are sometimes viewed as actions executed in business processes, as in [15] and [16]. Moreover, for different research needs, business activities sometimes require multiple views within one approach, e.g., [74], where they are viewed purely as actions of business logics in one example while viewed as compositions of actions of business logics and data in another example. Some approaches realize the modeling of business activities by viewing business activities as representations of different levels of granularity. For instance, in BPML [62], business activities are categorized into two types: complex activities and simple activities. A complex activity is an activity that contains one or more simple activities, establishes a context for their execution, and directs their execution by a definition of a specified activity set.

The identification and measurement of metrics of business process modeling has recently become an interest for researchers. A group of researchers have adapted the concepts of metrics in the discipline of software engineering and map them to be the needed metrics of business process modeling. For instance, a typical software metric, lines of code (LOC), is logically mapped to derive the “number of activities” in [75]. The authors of [76] derive a set of metrics for business processes after reifying the metrics of software processes. Moreover, some researchers propose metrics for evaluating certain non-functional requirements. A metric of weighted coupling is designed in [77] to evaluate the degree of coupling between different types of interrelations of business activities and processes. Metrics for evaluating the similarity between process models, are provided from the label-based, structural, and behavioral dimensions in [78]. IT-level metrics are favored to describe the artifacts and performance of IT systems in the realm of service-oriented business processes [79]: business-level, application-system-level, and the IT-infrastructure-level attributes are considered to be critical in representing the properties of the a networked-service, which is implemented by or implements other services of business processes.

Dependency among the metrics of business process modeling refers to the degree of one component of a business process relying on other components. The researchers of [80] generate one-to-one, one-to-many, and many-to-many dependencies to describe the degree of how objects and activities of business processes rely on each other. By exploring the dependencies between local business processes and global business processes, the authors of [81] define the metric inter-process dependencies and extend a UML-based activity diagram to all levels.

2.2 Business Process Optimization

Research of business process optimization can be generally categorized into optimization of business process modeling techniques [82–87] and optimization of business process information systems [88–99]. Scheduling is one major approach to optimize time-relevant properties in business processes. Researchers of [44] present an overview of mixed integer linear programming (MILP) based approaches to schedule chemical processing systems. By representing time in a discrete and continuous manner, they develop effective models for a variety of chemical processes and efficient solutions for difficult MILP models in a short-term scheduling domain. Using the similar time representation, researchers of [46] present two scheduling models: single-unit assignment models where task assignment is predetermined and multiple-unit assignment models where objects compete for processing products. Researchers of [47] aim at extending scheduling techniques of batch processes to handle large volume processes as well as different objectives. However, they are argued to be limitedly applied in the scenarios in which constraints cannot be mathematically modeled. Moreover, most of the techniques are NP-complete such that closed-form solutions cannot be provided by merely deriving those mathematical models.

Time sometimes is considered as a reference for modeling techniques. Researchers of [100] [39] include time as the dynamic view along with informational, functional, and organizational views together as the fundamental views to construct business processes. Assuming apriori knowledge of working time of each business activity, researchers of [1] present methods to detect the degree of parallelism, that is defined as the number of the running subprocesses at the same time. Researchers of [65, 66] implement empirical studies of software developments and try to identify the factors to cause the discrepancy of race time, that is defined as the time spent on actual work and elapsed time, that additionally includes the presence of interruption, blocking, and waiting periods. My approach, by analyzing the eligibility of how a business process can be optimized, provides methods to effectively optimize temporal performance for both a single subprocess and cluster of subprocesses.

2.3 Evaluation of Business Process Optimization

Ideally, evaluation of business process optimization refers to that given real-world business processes, their measured performance can be improved and optimized compared with that before applying the corresponding optimization approaches. To investigate the state-of-the-art of the research regarding evaluation of business process optimization, I conduct a paper survey by collecting 143 academic papers that were published from 2005 to 2015 in the mainstream academic conferences or journals and by generating a taxonomy for them. Generally, the major aspects of the evaluation of business process optimization are the evaluation of business process modeling techniques (e.g., [82–87]), the evaluation of business process information systems (e.g., [88–99]), the case studies of real-world business processes (e.g., [101–108]), the evaluation upon simulated business processes (e.g., [109–117]), the feedback from surveys (e.g., [118–120]), the non-numeral quality analysis (e.g., [121–124]), and no evaluations (e.g., [125–133]). The taxonomy results are demonstrated as Table 2.1.

Evaluation Types	Number of Papers	Ratio
Evaluation of Business Process Modeling Techniques	8	6%
Evaluation of Business Process Information Systems	13	9%
Case Studies of Real-World Business Processes	14	10%
Evaluation of Simulated Business Processes	22	15%
Feedback from Surveys	6	4%
Non-Numerical Quality Analysis	4	3%
No Evaluations	76	53%

Table 2.1: Paper survey results of aspects of evaluation for business process optimization

In this thesis, ultimately, the desired aspect of evaluation of business process optimization are the case studies of the real-world business processes. It can be observed that the portion of this type of evaluation aspect is small (10%) in the state-of-the-art of current research, according to the paper survey results shown in Table 2.1.

Another perspective towards the evaluation of the business process optimization is that whether the evaluation provides the comparison about the performance of the associated metrics before and after applying the evaluation techniques, or the comparison between the proposed evaluation techniques and the existing ones. I conduct surveys based on this perspective out of the papers with evaluation of business process optimization (the number of these papers is 143 - 76 = 67). The result is shown in Table 2.2.

Evaluation Types	Number of Papers	Ratio
Evaluation with Comparison	22	33%
Evaluation without Comparison	45	67%

Table 2.2: Paper survey results of evaluation w/o comparison for business process optimization

It can be easily observed from Table 2.2 that within the papers with e-valuation of their business process optimization approaches, only 33% of them compare the associated approaches with either the original cases before applying them or the existing relevant techniques.

Lastly, it is noted that only 5 out of 143 papers adopt the evaluation for their business process optimization upon real-world business processes with comparisons against either the circumstances before applying them or the associated existing relevant techniques.

The survey results indicate that the acquisition of business process resources has long been a major bottleneck in the research of business process optimization. Aiming at better selecting business process models, authors of [49] develop a web-based model that applies balanced scorecards, analytic hierarchy processes, and decision algorithms to approach an optimal solution. To improve the efficiency to model a business process, a recommendation method is proposed in [50] to mine the process repository and build a reference model such that its distance to the new model can be measured. Authors of [51] incorporate ontologies and knowledge maps to bridge the gap between business process models and the associated environments when optimizing business processes. Although these approaches can improve the efficiency of modeling business processes under certain constraints, they still assume the business processes that are used by their studies already exist.

2.4 Predictive Modeling for Business Management

Inventory forecasting has become a popular research topic recently. Beutel et al. [56] present data-driven frameworks to set safety stock by estimating demands using regression models and linear programming. Snyder et al. [57] examine different approaches to forecast demands of products that are intermittent and low volume. These approaches typically generalize inventory management scenarios and develop models accordingly, that are argued to be possibly limited in scalability and error-prone [134] [135] under varying scenarios.

Oracle EBS system [69] consists of a collection of enterprise resource planning (ERP), customer relationship management (CRM), and supply-chain management (SCM) applications. Oracle EBS system helps customers manage the complexities of global business environments for varying-size enterprises, such that customers can achieve immediate values and experience real-time information discoveries.

K Nearest Neighbors algorithms have been studied for a long time. Sankaranarayanan et al. [136] use R-tree to divide all data points into multiple blocks and build neighbor graph accordingly for faster calculations of distances. Similarly, in Zhang et al. [137] the technique of locality sensitive hashing is engaged with traditional KNN to divide items into small subsets with equal size, and then build one KNN graph on each subset using the brute force method. Assuming Gaussian distributions in [138], to find out the K nearest neighbors of a single data point, a certain number of data points are randomly selected for calculating distances and K smallest of them are grouped as the nearest neighbors that can be approached.

Chapter 3

Data-Centric Business Process Modeling

This chapter is based on my paper entitled "*A Goal-Directed Modeling Technique towards Business Process*" in the Proceedings of the 8th IEEE International Symposium of Service-Oriented System Engineering[1], coauthored by Professor Dewayne Perry, who supervised this paper and served as the correspondence author.

3.1 Summary

In this chapter, I study a data-centric business process modeling technique to explicitly model composite business activities that represent business processes with a well-defined syntax. Typically, a composite business activity is specified as a tuple of components: data, role, and atomic activity, where data is further modeled to indicate its states and lifecycle in data flows within a composite business activity. An atomic activity is defined as the activity which executes a single unit of work. That means partial data or activity generated within an atomic activity which does not impact on the service is not

[1] Yuqun Zhang and Dewayne E. Perry. A goal-directed modeling technique towards business process. In Service Oriented System Engineering (SOSE), 2014 IEEE 8th International Symposium on, pages 110–121, April 2014.

represented. Role refers to business actors executing or involved with atomic activities in the composite activity.

A workflow-based business process can be represented by the execution order of composite business activities. Specifically, a subprocess is defined to be composed of business activities in execution order. Subprocesses that are bounded by the identical convergence or divergence relations are defined as a cluster. A business process is configured by clusters and their placements.

3.2 Approach

A composite activity consists of the following components: data, role, and atomic_activity. Due to the states in data flows within a composite activity, data is further classified as initial_data, input_data, global_data, consumed_data, output_data, and final_data. The object declarations of each component and some examples are listed as follows:

```
type initial_data     primitive;
type input_data       primitive;
initial_data application_form;
input_data ready_to_start;
...
type global_data      primitive;
type consumed_data    primitive;
type output_data      primitive;
type final_data       primitive;
...
type role             primitive;
role applicant;
role government_employee;
...
type atomic_activity  primitive;
```

```
atomic_activity submit_application;
atomic_activity obtain_approval;
...
```

These object declarations, as in [61], include type definitions, type instances, and object definitions. Types and type definitions enable the appropriate abstractions and their values. The components are considered to be primitive and their objects are accordingly assigned. Note that data types are not limited only to data objects, but can also be states.

Each of the components of a composite business activity (abbreviated as 'activity' in the rest of the thesis) is defined and illustrated as follows.

3.2.1 initial_data

Definition 1. *An* initial_data *is the data that is initially injected to the data flows and triggers the execution of the associated activity along with* global_data *and* input_data *(if there is any).*

For any data, it is allowed to be an initial_data only once in each of its flows. An initial_data indicates the initial appearance of this data on the flow. In other words, this data is not included in the activities that are previously executed of the flow as any data type.

3.2.2 input_data

Definition 2. *An* input_data *is the data that is generated and delivered by other composite activities, and triggers the execution of the associated activity* *along with* global_data *and* initial_data *(if there is any).*

Theorem 1. *for any* input_data*, there must be at least an identical* output_data *in an activity that is executed beforehand, and vice versa.*

Similar to [139], since input_data marks the beginning of the associated execution of a composite activity, the output_data where it is delivered from must be located in a composite activity which has already been executed. It is possible that the composite activity associated with this output_data is not necessarily executed directly ahead of the composite activity associated with the identical input_data. In my approach, my rule is that an initial_data cannot be an input_data in the same activity.

3.2.3 global_data

Definition 3. *A* global_data *is the data that is injected independently from business processes, and, triggers the execution of the associated activity along with* initial_data *and* input_data *(if there is any).*

A global_data is not an output_data, final_data, or consumed_data of any activity, due to the fact that it is not the output of any activity. An example of global_data is "government ID" in the sample activity "loan application".

3.2.4 consumed_data

Definition 4. *A* consumed_data *is* input_data *or* initial_data *that is consumed by the execution of the associated activity.*

Consumed_data indicates the consumption of input_data or initial_data; it is not output_data in the associated composite activity. Consumed_data represents the termination as a data flow in the associated workflow.

3.2.5 output_data

Definition 5. *An* output_data *is the data that is the deliverable by the execution of the associated activity.*

As in Theorem 1, for an output_data, there is an identical input_data or initial_data in a composite activity that is executed later, unless it is a final_data. For any input_data that is not consumed in the associated activity, I generate an identical output_data in this activity.

3.2.6 final_data

Definition 6. *A* final_data *is the data that is generated by the execution of a business process and not consumed by any activity, or a state to indicate the termination of the execution of a business process.*

A final_data is represented only in a composite activity that is an end of a business process (there might be multiple activities that mark the termination of the execution of a business process). Note that one composite activity where an identical output_data is represented is not necessarily executed directly ahead of the activity associated with this final_data.

3.2.7 role

Definition 7. *A* role *is a human actor involved with the execution of an activity.*

A composite activity is executed by only one role or one specified group of roles.

3.2.8 atomic_activity

Definition 8. *An* atomic_activity *is the activity that executes a single unit of work within the associated activity.*

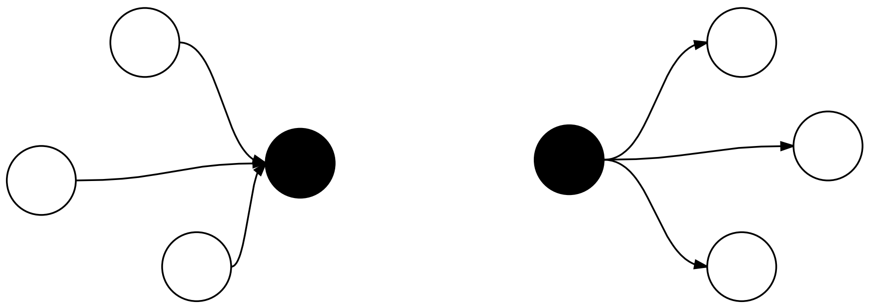
An atomic_activity cannot be further decomposed. Any data flow or human actor involvement that is included within an *atomic_activity* is not represented in the modeling of composite activities.

Overall, a composite business activity consists of one or more atomic activities that are executed under one role or one specified group along with a composition of initial_data, input_data, global_data, consumed_data, output_data, and final_data. For the sake of simplicity at this stage of my research, I do not address the issues involved in composite activities being components of composite activities and restrict my focus only atomic activities. It is reasonable because according to my activity structure, any composite activity inside of composite activities can be reduced to a composition of atomic activities, data, and human actors, and at last lead to my modeling of composite activities. In the rest of this thesis, I use this data-centric business modeling technique to represent their respective notions to maintain consistency.

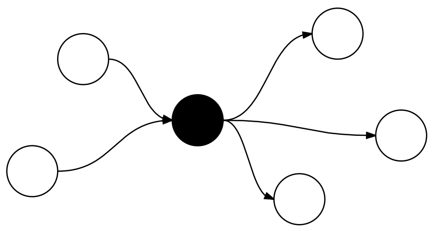
3.2.9 subprocess and cluster

Business processes can be complicated such that a single flow of control is not able to represent them. To represent this complexity, a subprocess is defined to be composed of composite business activities by their execution order. Subprocesses that are bounded by the identical convergence or divergence relations are defined as a cluster. A data-centric business process is configured by clusters and their placements.

A cluster can be designed as beginning_cluster, join_cluster, split_cluster, join_and_split_cluster, or singular_cluster, due to the number of the executed clusters that triggers it and/or the number of the clusters that its execution triggers. Beginning_cluster denotes that the execution of a cluster is triggered by no other clusters and triggers the execution of other clusters. Join_cluster indicates that the execution of a cluster is triggered by the execution of multiple clusters. Split_cluster indicates that the execution of a cluster triggers the execution of multiple clusters. Join_and_split_cluster indicates that the execution of a cluster is triggered by and triggers the execution of multiple clusters.



(a) *join_cluster* (b) *split_cluster*



(c) *join_and_split_cluster*

Figure 3.1: Examples of join_cluster, split_cluster, and join_and_split_cluster, that are represented by black dots.

Singular_cluster indicates the execution of a cluster is triggered by and triggers the execution of only one or none cluster, Some examples of join_cluster, split_cluster, and join_and_split_cluster are listed in Figure 3.1 for better illustration.

3.3 An Example

To better illustrate the data-centric business process modeling technique, I use an example of a business process of "purchasing apartments" that was originally modeled by BPMN [1, 2] as in Figure 3.2 and model it to be a data-centric business process.

One of its activities "loan application" and the illustration of its components is stated as follows:

```
activity  loan_application
   {
   role  applicant, bank_employee, bank_manager;
   initial_data  loan_application_form;
   input_data empty;
   global_data  government_ID,
                bank_statement,
                credit_history;
   consumed_data  loan_application_from;
   atomic_activity   <fill_out_loan_application_form>,
                     <submit_application_form>;
   output_data  approved_loan_application_form,
   final_data  empty;
   }
```

In this activity, the roles are "applicant, bank_employee, bank_manager". "Loan_application_form" is the initial_data that is consumed by executing this activity. That means this data would not play any role of the rest business process. "Government_ID", etc., are the global_data whose states are not impacted by the execution of any activity in the business process. By setting "fill_out_loan_application_form" and "submit_application_form" as the atomic activities, no data other than "approved_loan_application_form" is the

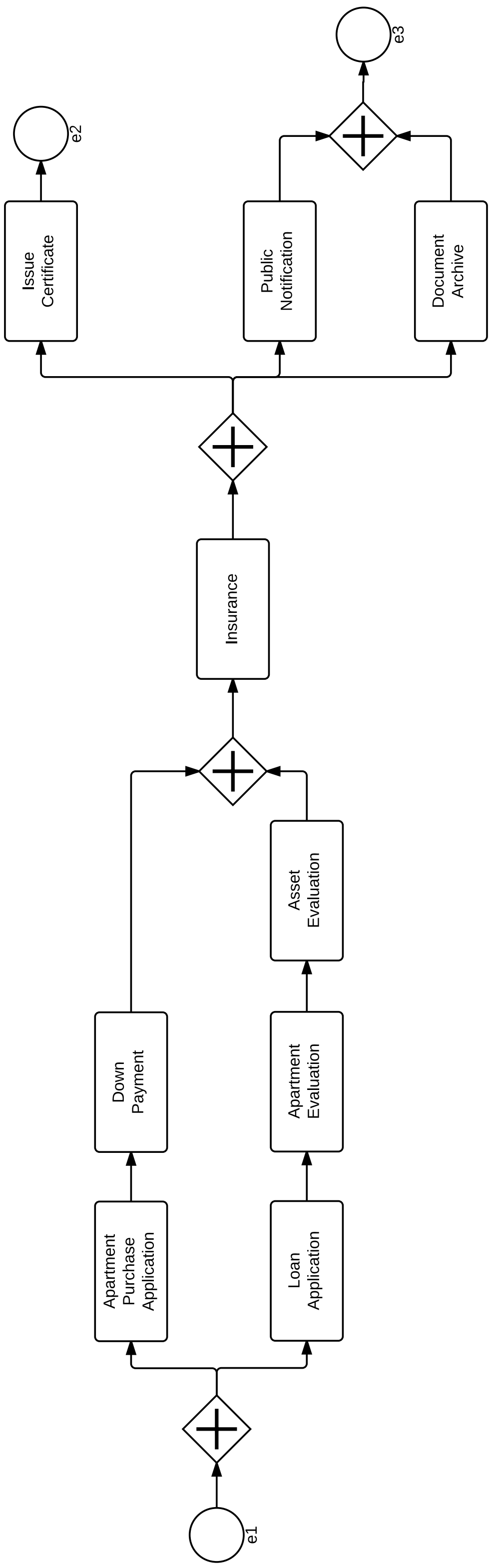


Figure 3.2: The original BPMN model of purchasing apartments

ouput_data of this activity that is delivered as the input_data of the rest activities. Note that “approved_loan_application_form” can be understood as a updated state of the input_data “loan_application_form” after executing this activity.

The entire example is demonstrated in Appendix A.

Chapter 4

Temporal Performance Optimization upon Data-Centric Business Processes

This chapter is based on my paper entitled "A Data-Centric Approach to Optimize Time in Workflow-Based Business Process" in the Proceedings of the 11th IEEE International Conference on Services Computing[1], coauthored by Professor Dewayne Perry, who supervised this paper and served as the correspondence author.

4.1 Summary

In this thesis, the working time of business process is calculated by combining the working time of the associated activities. Business processes can be complicated such that multiple flows of control might be included. Therefore, execution order of activities in different flows can be reflected by their partial order. While changing the execution order of activities is possible in different flows, the working time of the business processes is possible to be changed accordingly. On the other hand, changing the execution order of

[1]Yuqun Zhang and Dewayne E. Perry. A data-centric approach to optimize time in workflow-based business process. In Services Computing (SCC), 2014 IEEE International Conference on, pages 709–716, June 2014.

activities can be risky because workflow of business processes are possible to be undermined. Therefore, to maintain the validity of business process workflow, the associated constraints need to be detected and can be represented by the partial order of activities and their data transmissions.

In this chapter I study an approach to optimize temporal performance of the data-centric business processes, that is classified to two stages: the detection of the eligibility for business activity relocation and the algorithms of temporal optimization. Typically, my approach is initiated by identifying whether one business activity is eligible to be relocated to a given subprocess. With the knowledge of the eligibility and the working time of each business activity, my algorithm selects proper eligible activities for the subprocess to approach an optimal temporal performance. Accordingly, the temporal optimization of one cluster is approached by modifying the criteria of selecting the eligible activities and effectively assigning them to their eligibly-relocated subprocesses.

4.2 Approach

4.2.1 Eligibility for Business Activity Relocation

4.2.1.1 Motivation

A data-centric business process of an ideal temporal performance is designed with deterministic execution order of business activities, that is, any change of the execution order of the business activities can increase the working time of the business processes. However, business processes can be updated frequently because of the various reasons, such as the updates of the business objectives, resources, etc. For instance, in Figure 3.2, assume the input data of the activity "insurance" is loosened to be "approved_loan_application_form or approved_apartment_application_form" instead of "approved_loan_application-_form and approved_apartment_application_form", then this business process can be updated by executing "insurance" in either subprocess, as illustrated in Figure 4.1.

Usually in time studies of business processes, the working time for each activity is assumed to be given [1]. In this thesis, to optimize the temporal performance of business process given the updates of Figure 4.1 is equivalent to find out which possible business process, Figure 3.2 or Figure 4.1, gives the best temporal performance.

4.2.1.2 An Analogous Representation

Pipe-and-filter (PnF) systems are one of the fundamental architecture styles used in component based software engineering [140] [141] [142]. A pipe-and-filter (PnF) graph is multigraph, where components are represented by boxes and connectors are represented by pipes. Data is passed between boxes via pipes. Boxes may receive inputs and produce outputs, where input ports are drawn as nubs on their left sides and output ports are drawn as nubs on their right sides. A connector links input ports to output ports.

To better illustrate the possibility of how business processes can be updated, PnF graphs are adapted and used in this thesis. Particularly, the

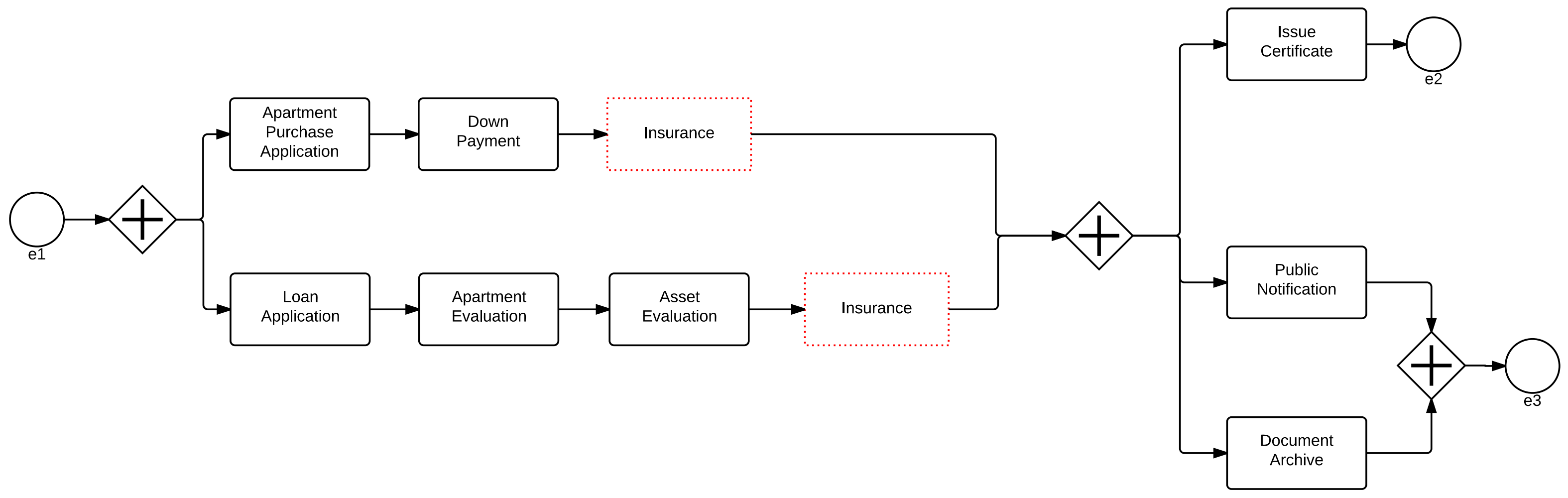


Figure 4.1: The BPMN model of purchasing apartments updated by modifying the execution of insurance

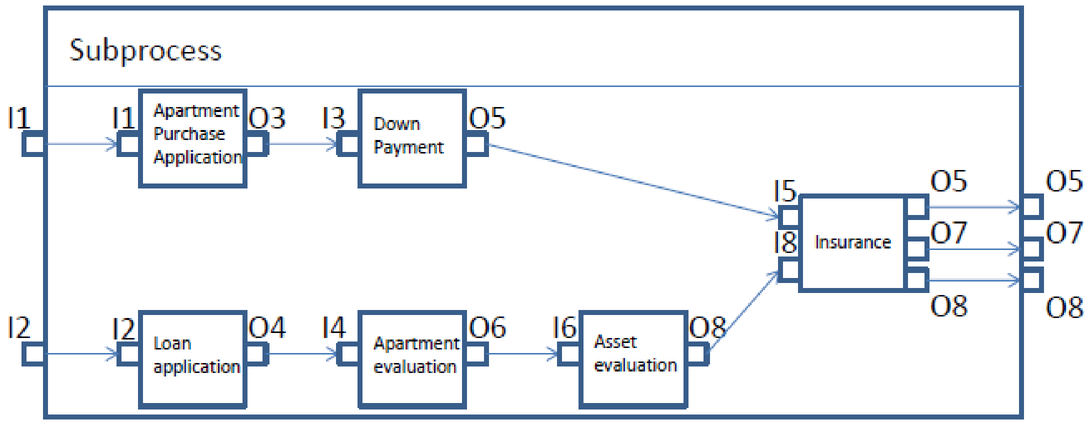


Figure 4.2: The PnF representation of "apartment purchasing" process

subprocesses from the beginning of the "apartment purchase" process to the activity "insurance" of Figure 3.2 could be represented as Figure 4.2.

For brevity, the inputs and outputs of the activities in Figure 4.2 are symbolized and simplified. For instance, the activity "Insurance" is executed when its two inputs "I5" and "I8", that are passed by "O5" and "O8" of the activity "Down Payment" and "Asset Evaluation" respectively, are both available. Its outputs are "O5", "O8", and "O7".

Assume at some point when running this business process, the policies are modified such that the inputs of the activity "Insurance" are changed from "I5" and "I8" to be "I5" or "I8", that is shown in Figure 4.3.

It is apparently that Figure 4.3 can be rewritten as Figure 4.4(a) or Figure 4.4(b).

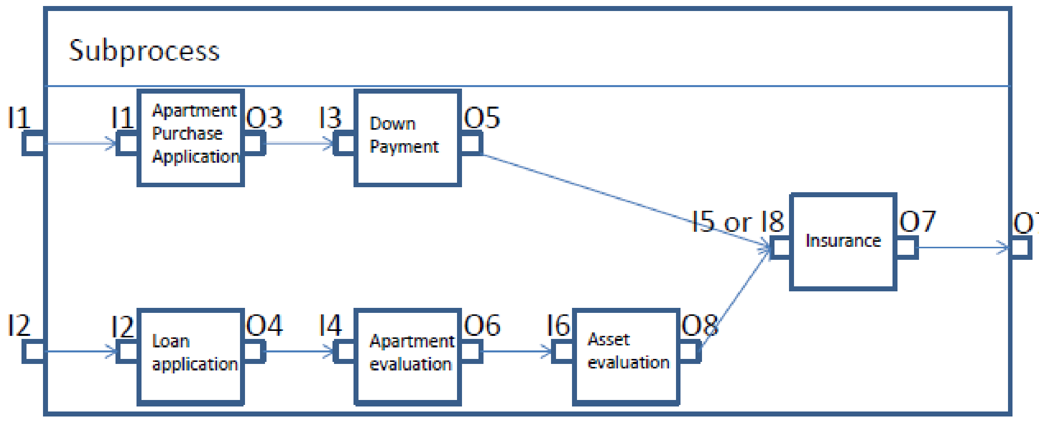
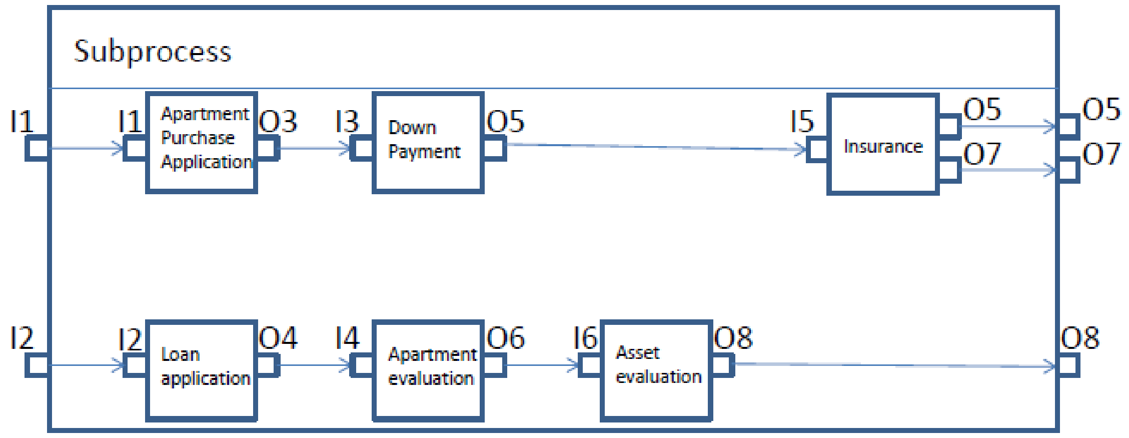


Figure 4.3: The PnF representation of the updated "apartment purchasing" process

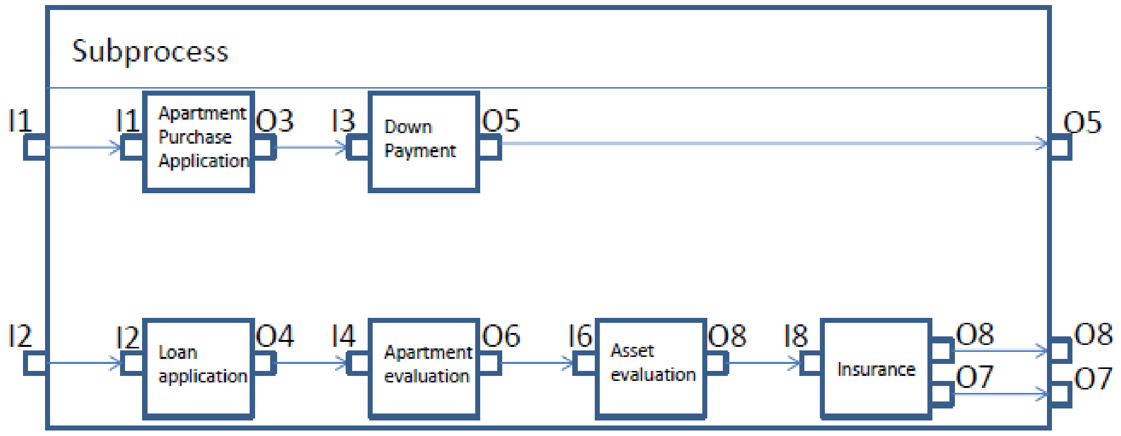
4.2.1.3 My Solutions

It can be derived from above that, in reality, it is possible for an optimized-designed business processes being updated to be not optimized; on the other hand, it is possible that a business activity can be flexibly executed if it is not directly dependent on the execution of its adjacently-previous or subsequent activities.

My approach specifies this possibility by defining that for any execution of adjacent business activities, the input_data of the later-executed activity is not necessarily the output_data of the former-executed activity. Specifically, assume a business activity, namely BA, with its input_data being part of the output_data generated only by a subprocess SP_A that is executed earlier but not adjacently ahead. Then the advanced execution of BA adjacently following



(a)



(b)

Figure 4.4: The two possible updated "apartment purchasing" processes

SP_A is legitimate (assuming no other sources of impact), because this flexible execution does not impact on their original order of data transmission that ensures the execution of the business process. Analogously, if the output_data of BA is part of the output_data of a subprocess SP_B only and SP_B is executed not adjacently later, then the postponed execution of BA adjacently ahead of SP_B is legitimate as well. This observation provides the fundamentals for flexible execution of business activities in business process, that is realized by applying the data-centric approach.

This reflection of the execution order of business activities provides a possibility of deriving a partial order of any given subprocess and activity. Define *TP(A)* to denote the array of activities ahead of activity A , and *TP(SP)* to denote the array of activities ahead of subprocess SP in a *TP*. The partial order between A and SP can be derived by comparing the elements in *TP(A)* and *TP(SP)*.

Theorem 2. *Activity A is* ***executed before*** *subprocess SP when* TP(SP) $\subset$ TP(A)*, and Activity A is* ***executed after*** *subprocess SP when* TP(A) $\subset$ TP(SP)*.*

Any *TP* that is a subset of another one indicates their corresponding activity/subprocess are in the same workflow, and their execution order can be completely identified. For instance, in Figure 3.2, given the subprocess {Loan Application, Apartment Evaluation, Asset Evaluation} and the activity ”Insurance”, the activity “Insurance” is executed after the subprocess since the

TP of the subprocess is the subset of the *TP* of the activity, that is {Loan Application, Apartment Evaluation, Asset Evaluation, Insurance}.

Theorem 3. *Activity A is* ***loosely parallel*** *to subprocess SP when* TP(SP) $\not\subset$ TP(A), *and* TP(A) $\not\subset$ TP(SP).

As opposed to Theorem 2, The fact that the *TP*s of a pair of activity/subprocess are not mutually subsets of one another indicates disjoint workflows between them. This parallelism is considered to be loose because the chronology of A and SP can actually be detected by deriving the total order of activities/subprocesses based on the properties of the data-centric approach and the knowledge of their working time, yet it is not necessary in my approach at this point. For instance, given the activity "Loan Application" and the subprocess {Apartment Purchase Application, Down Payment}, the activity is loosely parallel to the subprocess since the *TP* of the subprocess {Apartment Purchase Application, Down Payment} is not a subset of the *TP* of the activity {Loan Application}, and vice versa.

In the data-centric approach, a workflow-based business process is perceived as being established and maintained by its dataflow, i.e., the execution of a business process is realized by the data delivery among business activities. Hence to explore the eligibility of relocating a business activity of a business process is essentially to find out whether the data delivery in the business process would be impeded after relocating the business activity. Assuming in my approach at this point that a business activity is kept consistent without modifying its components (i.e., data types, human actors, and atomic activities), data delivery is ensured to be safe when its pattern, specifically the order of data transmission among business activities, is kept identical with that in the original workflows. Particularly, for any relocated activity, the data delivery of a business process is secured when the input_data of that activity is part of the data that is delivered by all its former-executed activities (e.g., the output_data that has not been consumed), and the output_data of that activity is part of the data that is delivered to all its later-executed activities (e.g., the input_data) in its workflow.

Definition 9. S_{target} *denotes a partial subprocess.* A_{target} *denotes an activity that is not part of* S_{target}*.* $\mathrm{S}_{inbetween}$ *denotes all the partial subprocesses between* S_{target} *and* A_{target}*.* $\mathrm{Input}_{A_{target}}$*,* $\mathrm{output}_{A_{target}}$*,* $\mathrm{Initial}_{A_{target}}$*, and* $\mathrm{consumed}_{A_{target}}$ *respectively denote the sets of all the* input_data*,* output_data*,* initial_data*, and* consumed_data *of* A_{target}*.* $\mathrm{S}_{beforetarget}$ *denotes the set of the subprocesses that are* ***executed before*** S_{target} *(including* S_{target}*) in its workflow.* $\mathrm{Input}_{S_{beforetarget}}$*,* $\mathrm{output}_{S_{beforetarget}}$*,* $\mathrm{initial}_{S_{beforetarget}}$*, and* $\mathrm{consumed}_{S_{beforetarget}}$ *respectively denote the sets of the* input_data*,* output_data*,* initial_data*, and* consumed_data *of all the activities of* $\mathrm{S}_{beforetarget}$*.* $\mathrm{input}_{S_{inbetween}}$ *denotes the set of* input_data *of* $\mathrm{S}_{inbetween}$*.*

$S_{beforetarget}$ can be derived by searching the *TP*s that are contained in *TP(S_{target})*. $S_{inbetween}$ can be derived by searching the *TP*s that are contained in *TP(S_{target})* and *TP(A_{target})* and identifying their differences.

Theorem 4. A_{target} *is* ***backwardly moveable*** *to* S_{target} *when 1)* A_{target} *is* ***executed after*** S_{target}*; 2)* $\mathrm{input}_{A_{target}}$ *only belongs to (*$\mathrm{input}_{S_{beforetarget}} \cup \mathrm{initial}_{S_{beforetarget}} \cup \mathrm{output}_{S_{beforetarget}}$*)* - $\mathrm{consumed}_{S_{beforetarget}}$*, and none of the* $\mathrm{consumed}_{A_{target}}$ *belongs to* $\mathrm{input}_{S_{inbetween}}$.

Theorem 4 illustrates under what circumstances the execution of A_{target} is eligible to be brought forward to be adjacently following the execution of S_{target}. According to the data-centric modeling technique, it is possible that in $S_{beforetarget}$, a single piece of data is transmitted among different activities as multiple data types. Therefore, ($\mathrm{input}_{S_{beforetarget}} \cup \mathrm{initial}_{S_{beforetarget}} \cup \mathrm{output}_{S_{beforetarget}}$) - $\mathrm{consumed}_{S_{beforetarget}}$ ensures accurate data delivered from $S_{beforetarget}$. Since A_{target} is brought forward, the impact of its output_data delivery to the subsequent subprocesses stays consistent and is not necessarily cared about.

In addition, to ensure that the input_data of any of the $S_{inbetween}$ is not consumed by the execution by A_{target}, elements of $\mathrm{consumed}_{A_{target}}$ need to be ensured by not belonging to $input_{S_{inbetween}}$, such that the data delivered from the previous executed activities are not consumed in advance of the subprocesses of $S_{beforetarget}$ that still need them for execution.

Figure 3.2 is used here to illustrate how Theorem 4 works. Assume A_{target} as "Insurance" and S_{target} as {Apartment Purchase Application, Down Payment}. Accordingly, the sets of different data types are listed in Table 4.1.

$output_{S_{beforetarget}}$	approved_apartment_application_form,
$consumed_{S_{beforetarget}}$	apartment_application_form, contract,
$input_{S_{beforetarget}}$	approved_apartment_application_form,
$initial_{S_{beforetarget}}$	apartment_application_form, contract
$input_{A_{target}}$	approved_apartment_application_form or approved_loan_application_form,
$consumed_{A_{target}}$	empty

Table 4.1: Sets of data types

The computation result of (input$_{S_{beforetarget}}$ ∪ initial$_{S_{beforetarget}}$ ∪ output$_{S_{beforetarget}}$) - consumed$_{S_{beforetarget}}$ is {approved_apartment_application_form} that are the input_data of the updated activity "Insurance". Moreover, the affect of the relocation of A_{target} does not change the way that consumed$_{A_{target}}$ works. Therefore, the activity "Insurance" is eligible to be relocated to the subprocess {Apartment Purchase Application, Down Payment}.

Definition 10. S$_{aftertarget}$ *denotes the set of the subprocesses that are* ***executed after*** S$_{target}$ *(including* S$_{target}$*) in its workflow.* S$_{inbetween'}$ *denotes all the partial subprocesses between* S$_{target}$ *and* A$_{target}$*.* Input$_{S_{aftertarget}}$*,* output$_{S_{aftertarget}}$*,* initial$_{S_{aftertarget}}$*, and* consumed$_{S_{aftertarget}}$ *respectively denote the sets of the* input_data*,* output_data*,* initial_data*, and* consumed_data *of all the activities of* S$_{aftertarget}$*.*

As opposed to the way of deriving $S_{beforetarget}$, $S_{aftertarget}$ can be derived by searching the partial *TP*s with S_{target} as the beginning. $S_{inbetween'}$ can be derived by searching the partial *TP*s with S_{target} and A_{target} as the beginning and identifying their differences.

Theorem 5. A_{target} *is **forwardly moveable** to* S_{target} *when 1)* A_{target} *is **executed before*** S_{target}*; 2) none of* $\mathrm{output}_{A_{target}}$ - *(*$\mathrm{output}_{A_{target}} \cap \mathrm{input}_{A_{target}}$*) belongs to* $\mathrm{input}_{S_{inbetween'}}$.

Theorem 5 illustrates under what circumstances the execution of A_{target} is eligible to be postponed to the execution of S_{target}. Since executing A_{target} is postponed, the impact from its former-executed subprocesses to its input_data stay consistent and is not my concern. In addition, it is necessary to make sure that the postponed execution of A_{target} does not damage the workflow of $\mathrm{input}_{S_{inbetween'}}$, that can be reflected by identifying the newly generated ouput_data after executing A_{target} is not the input_data of any partial subprocess of $\mathrm{input}_{S_{inbetween'}}$.

Note that in this thesis, the circumstances when an activity is originally executed after a subprocess and relocated before it or is originally executed before a subprocess and relocated after it are not considered, because essentially theses circumstances are equal to the ones above. For instance, for the circumstance of activity being relocated before a subprocess when it is originally executed after that subprocess, it is usually equivalent to be relocated before another subprocess that is executed after the target subprocess, and vice versa. Therefore, the solutions are the same as discussed above.

To explore the pattern of a business activity being relocated when it is loosely parallel to a subprocess, there are two fundamental scenarios that need to be considered: *1)* the activity would be executed after the subprocess; *2)* the activity would be executed before the subprocess. Theorem 6 illustrates the pattern when an activity is executed after a subprocess.

Theorem 6. A_{target} *is* ***loosely-parallelled moveable*** *to* S_{target} *when 1)* A_{target} *is* ***loosely parallel*** *to* S_{target}*; 2)* $\mathrm{input}_{A_{target}}$ *belongs to (*$\mathrm{input}_{S_{beforetarget}}$ $\cup$ $\mathrm{initial}_{S_{beforetarget}}$ $\cup$ $\mathrm{output}_{S_{beforetarget}}$*)* - $\mathrm{consumed}_{S_{beforetarget}}$*, and none of the* $\mathrm{consumed}_{A_{target}}$ *belongs to any of* $\mathrm{input}_{S_{aftertarget}}$*; 3) none of* $\mathrm{output}_{A_{target}}$ - *(*$\mathrm{output}_{A_{target}}$ $\cap$ $\mathrm{input}_{A_{target}}$*) belongs to the* input_data *of any of the partial subprocesses that are* ***executed after*** *the subprocess that* A_{target} *is originally located in.*

Theorem 7 illustrates the pattern when an activity is executed before a subprocess.

Theorem 7. A_{target} *is* ***loosely-parallelled moveable*** *to* S_{target} *when 1)* A_{target} *is* ***loosely parallel*** *to* S_{target}*; 2)* $\mathrm{input}_{A_{target}}$ *belongs to (*$\mathrm{input}_{S_{beforetarget}+S_{target}}$ $\cup$ $\mathrm{initial}_{S_{beforetarget}+S_{target}}$ $\cup$ $\mathrm{output}_{S_{beforetarget}+S_{target}}$*)* - $\mathrm{consumed}_{S_{beforetarget}+S_{target}}$*, and none of the* $\mathrm{consumed}_{A_{target}}$ *belongs to any of* $\mathrm{input}_{S_{aftertarget}+S_{target}}$*; 3) none of* $\mathrm{output}_{A_{target}}$ - *(*$\mathrm{output}_{A_{target}}$ $\cap$ $\mathrm{input}_{A_{target}}$*) belongs to the* input_data *of any of the partial subprocesses that are* ***executed after*** *the subprocess that* A_{target} *is originally located in.*

Obviously, the conditions to trigger a **loosely-parallelled moveable** business activity are more constrained compared with those of Theorem 5 and 4.

In summary, given a subprocess and an activity, the activity is eligible for relocation in the business process when it can be **forwardly moveable**, **backwardly moveable**, or **loosely-parallelled moveable** to a different subprocess.

4.3 Algorithms of Temporal Optimization

In this thesis, temporal inefficiency is considered to be caused by the waiting time of subprocesses, that is generated under the circumstances that their executions are completed while other subprocesses in the same cluster are still being executed. Given the apriori knowledge of the working time of each activity of business process, the working time of a subprocess can be calculated by simply summing up the working time of all its activities. Then the waiting time of a subprocess is derived by subtracting the working time of the cluster by the working time of this subprocess. Note in my approach, working time and waiting time are considered as worst-case metrics. Particularly, the working time of a cluster is defined as the largest working time of all its subprocesses, regardless their convergence/divergence relations. Therefore, even though in a cluster that is bounded by XOR relations where the subprocess with the shortest time is normally executed, the working time of this cluster is not simply calculated as the shortest working time among its subprocesses.

4.3.1 Temporal Optimization for A Single Subprocess

It is possible in reality that only one subprocess needs to optimize its temporal performance to achieve its associated business goal(s) without taking the overall temporal performance of the cluster or overall business process into account. In my approach, to optimize temporal performance for a single subprocess, namely S_{target} is simply equivalent to optimize its working time by filling itself with executing additional eligible activities from other subprocesses.

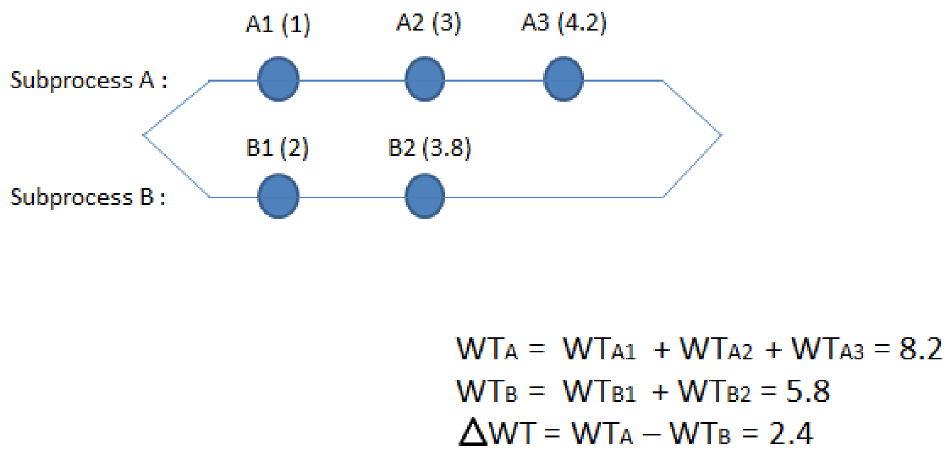
Each activity in the business process is detected beforehand whether they are eligible for S_{target}, that is, whether they are **forwardly moveable**, **backwardly moveable**, or **loosely-parallelled moveable** to S_{target}. There might be multiple eligible activities, namely $A_{eligible}$s in other subprocesses, all of which could contribute to optimizing waiting time of S_{target}. Ideally, they are considered to be selected and relocated together to S_{target} to minimize the working time. However, it is only allowed to have one $A_{eligible}$ relocated to S_{target} at one time, because after any activity is relocated to S_{target}, the patterns of data delivery of S_{target} need to be updated accordingly. It is possible that multiple $A_{eligible}$s might be "incompatible" with each other's workflow, such that data delivery of S_{target} would be impeded if they are relocated together to S_{target}.

Furthermore, this brings up a typical NP-hard problem. Each time when any $A_{eligible}$ is relocated to S_{target}, it is necessary for S_{target} to update the eligibility of all the activities and reselect the eligible activities. After an

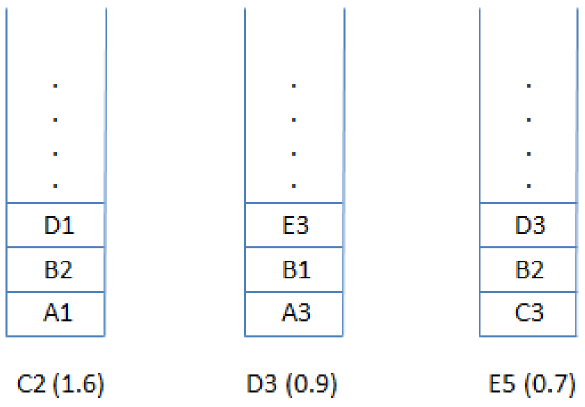
$A_{eligible}$ is relocated to S_{target}, the subsequent choice of $A_{eligible}$ cannot guarantee the optimal accumulative temporal performance compared with combination of other $A_{eligible}$s. Under the constraints of my approach, the optimal waiting time cannot be obtained unless all the possible $A_{eligible}$ combination are tried out and each of their performance relative to waiting time optimization is compared. Even given a solution of $A_{eligible}$ combination, the only way to find out whether it is optimal is still to enumerate all the possible $A_{eligible}$ combinations and compare all the results in my approach at this point. This can be generalized that the optimality of a given solution of this problem cannot be ensured by any method to my knowledge within polynomial time. Therefore, to optimize time performance for a single subprocess is an NP-hard problem under the constraints of this approach.

To approach an optimal solution in this case, techniques of approximation need to be adopted in my algorithms. Initially, the absolute value of the contribution to time optimization from each $A_{eligible}$, namely AB_{time} is calculated as the waiting time of S_{target}, namely WA_{target} subtracting the working time of each $A_{eligible}$, namely $WO_{eligible}$. $A_{eligible}$s are sorted in an ascending order of their AB_{time}.

After the first element in $\{A_{eligible}\}$ is extracted, $\{A_{eligible}\}$, along with $\{WA_{target}\}$ and $\{AB_{time}\}$ are updated according to the varying pattern of the data delivery of S_{target}. After sorting $\{AB_{time}\}$, the smallest one is compared with WA_{target}. If it is larger than WA_{target}, then WA_{target} is believed to be the optimal waiting time of S_{target} and no further update is needed. Otherwise the above process is iterated until a smaller WA_{target} is found. This algorithm, instead of enumerating all the possibilities, affirmatively selects the observed optimal solutions to approach an approximately optimal waiting time of a single subprocess.



(a) *Working time of subprocess A and B*



(b) *Eligible activities to subprocesses A and B*

Figure 4.5: An example of a cluster with two subprocesses and the activities eligible to be relocated to them

The pseudo code of this algorithm is listed in Algorithm 1. In my approach, the details of sorting and activity path discovery are not addressed because they are well-known sophisticated techniques.

Algorithm 1 Optimize_Time (Cluster_Subprocess, Waiting_Time): deriving the optimized time for a cluster subprocess

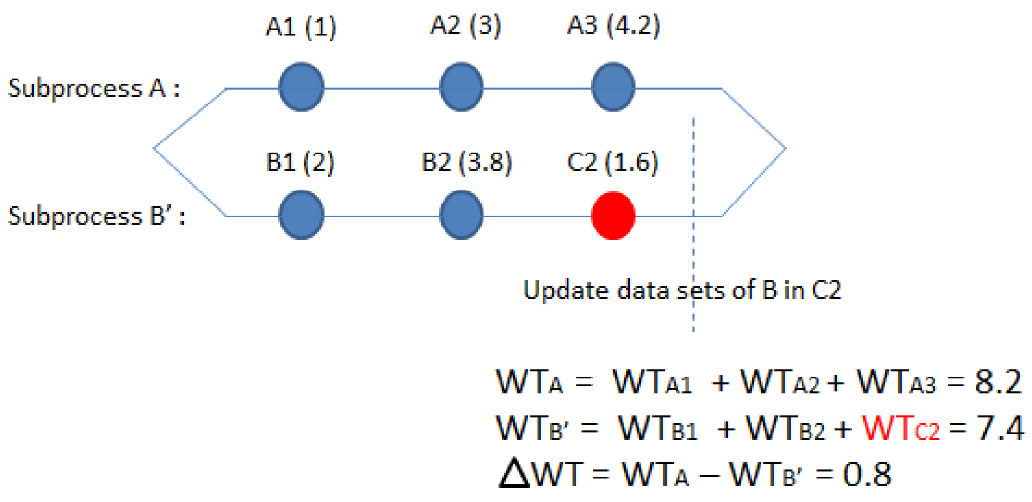
```
Input:
$S_{target}$ := a cluster process that needs to optimize time performance,
$A_{eligible}$ := activity that is eligible for relocation to the cluster process that needs to optimize time performance,
$WO_{cluster}$ := working time of the cluster, a constant
$WA_{target}$ := waiting time of the cluster process that needs to optimize time performance,
$WO_{target}$ := working time of the cluster process that needs to optimize time performance,
$WO_{eligible}$ := working time of the eligible activities,
$AB_{time}$ := absolute value of working time of the eligible activities subtracting waiting time of the cluster process that needs to be timely optimized,
Output:
$Optimize\_Time(S_{target}, WA_{target})$ := optimized waiting time of $S_{target}$

1: $WA_{target}$ := absolute value of ($WO_{cluster}$ - $WO_{target}$);
2: $\{A_{eligible}\} := \emptyset$;
3: $\{WO_{eligible}\} := \emptyset$;
4: $\{AB_{time}\} := \emptyset$;
5: for each $A \notin S_{target}$ in business process do
6:     check whether $A$ is forwardly moveable, backwardly moveable, or loosely-parallelled moveable to $S_{target}$;
7:     if yes then
8:         $\{A_{eligible}\} := \{A_{eligible}\} + A$;
9:         $AB_{time}$ := absolute value of ($WA_{target}$ - $WO_{eligible}$);
10:        $\{AB_{time}\} := \{AB_{time}\} + AB_{time}$;
11:        $\{WO_{eligible}\} := \{WO_{eligible}\} + WO_{eligible}$;
12:    end if
13: end for
14: sort($\{AB_{time}\}$);
15: sort($\{WO_{eligible}\}$) according to sort($\{AB_{time}\}$);
16: sort($\{A_{eligible}\}$) according to sort($\{AB_{time}\}$);
17: if $\{AB_{time}\}[0] < WA_{target}$ and $\{A_{eligible}\} \neq \emptyset$ then
18:     $WO_{target} := WO_{target} + \{WO_{eligible}\}[0]$;
19:     $S_{target} := S_{target} + \{A_{eligible}\}[0]$;
20:     $Optimize\_Time(S_{target}, WA_{target})$;
21: else
22:     break;
23: end if
```

An example is used to illustrate how the algorithm above is applied in a simple scenario, as in Figure 4.5, where Figure 4.5(a) introduces a cluster that is composed of two subprocesses A and B, and Figure 4.5(b) lists all the activities that are eligible to be relocated to other subprocesses along with these possible partial subprocesses. In Figure 4.5(a), the waiting time of subprocess B is 2.4. Since the working time of activity C2 is the largest among all the eligible activities, relocating C2 can result in the maximum reduction of the waiting time of subprocess B. Therefore, C2 is relocated to subprocess B with the waiting time of subprocess B updated to be 0.8, and the corresponding data types updated, as in Figure 4.6(a). In addition, after C2 being relocated to subprocess B, the eligible activities are subsequently updated by deleting C2, as in Figure 4.6(b).

The process above is repeated for the remaining eligible activities D3 and E5. It can be observed that relocating E5 to subprocess B can reduce its waiting time by 0.7. Therefore, E5 is relocated to subprocess B and the waiting time of subprocess B is updated to be 0.1, as in Fig 4.7. Assuming at this time D3 is still eligible to be relocated to subprocess B, it would not be relocated to subprocess B since relocating it would result in the increasing waiting time of the subprocess A. That terminates the process to reduce the waiting time for a single subprocess.

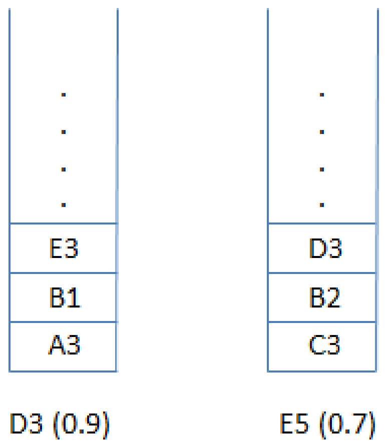
4.3.2 Temporal Optimization for One Cluster

In some business processes, a cluster implies one or one group of business object(s) where its components are likely to be tightly relevant to each other. Therefore, it can be beneficial to reduce the working time of one cluster within the business process. In this thesis, the technique of optimizing the



$$WT_A = WT_{A1} + WT_{A2} + WT_{A3} = 8.2$$
$$WT_{B'} = WT_{B1} + WT_{B2} + WT_{C2} = 7.4$$
$$\Delta WT = WT_A - WT_{B'} = 0.8$$

(a) *Updated working time of subprocess A and B*



(b) *Updated eligible activities to subprocesses A and B*

Figure 4.6: The updated cluster with activity C2 relocated to the subprocess B and updated eligible activities accordingly

working time of one cluster is generally based on that of a single subprocess. Instead of focusing on simply optimizing the waiting time in one single subpro-

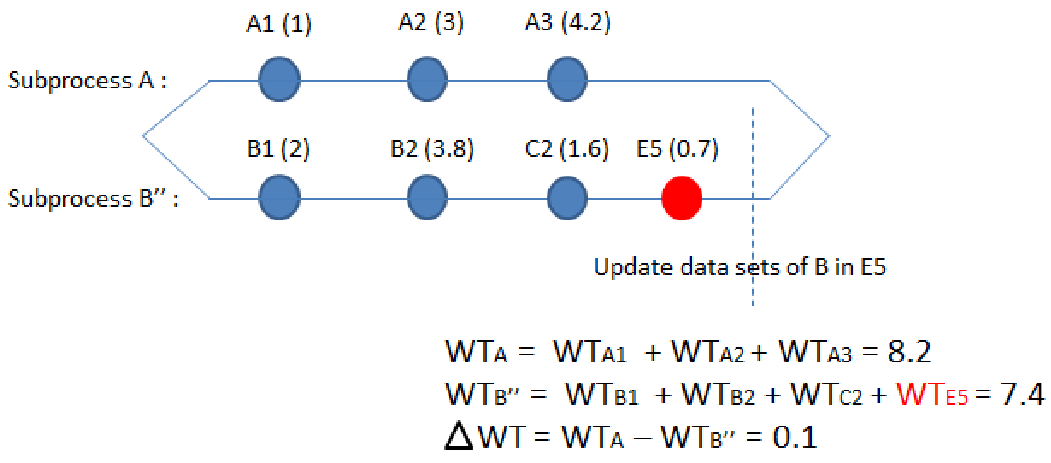


Figure 4.7: The final updated cluster with the waiting time of subprocess B minimized

cess, the technique of temporal optimization for one cluster aims at reducing average working time of clusters.

One major difference is the fact that AB_{time}, that is calculated simply as the absolute value of WA_{target} subtracting $WO_{eligible}$, cannot be used as the criteria to accurately reflect the overall time optimization. In fact, not only the contribution from $WO_{eligible}$ after $A_{eligible}$ being relocated to S_{target}, but also the impact that the relocation of $A_{eligible}$ imposes on its original subprocess needs to be considered. Define $VA_{original}$ as the varying working time of the cluster where $A_{eligible}$ is originally located. $VA_{original}$ is calculated as the working time of the original cluster before $A_{eligible}$ being relocated, subtracting the working time of the original cluster after $A_{eligible}$ being relocated. Similarly, VA_{target}

denotes the varying working time of the cluster that $A_{eligible}$ is relocated to and is calculated as the working time of that cluster after $A_{eligible}$ being relocated subtracting the working time of that cluster before $A_{eligible}$ being relocated.

VA_{sum}, defined as the summation of $VA_{original}$ and VA_{target}, is used to indicate how the relocation of $A_{eligible}$ changes the total working time of the clusters that it is relocated from/to. If VA_{sum} is negative, then the total working time of the clusters that $A_{eligible}$ is relocated from/to is decreased, and vice versa.

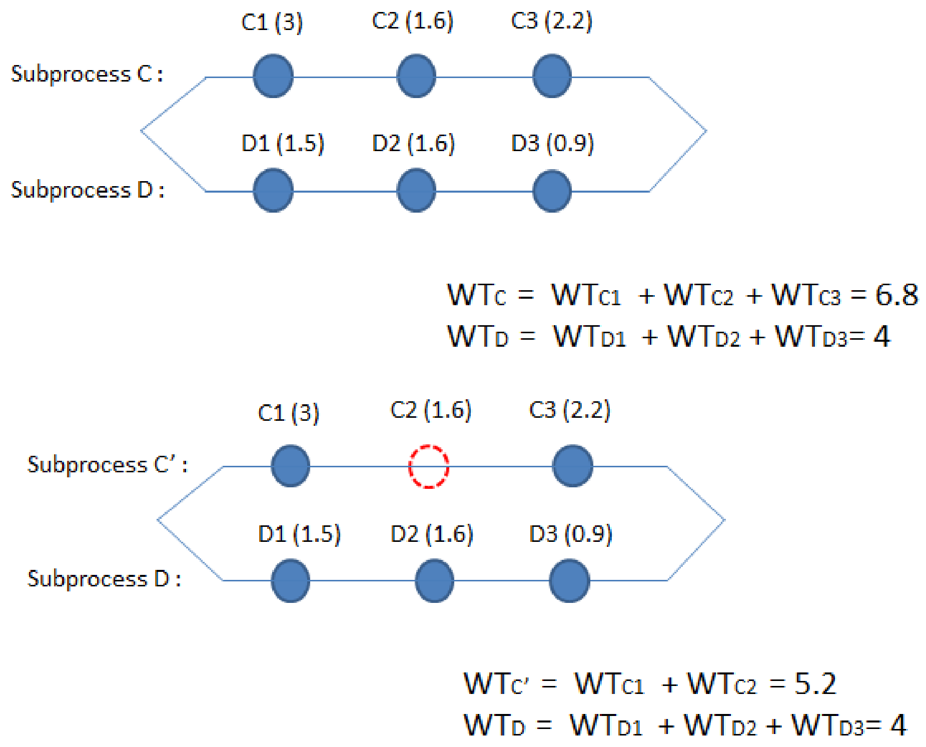


Figure 4.8: Updated working time of subprocess C after C2 being relocated

Ideally, to achieve the temporal optimization of overall business process, it is equivalent to realizing the temporal optimization of each subprocess and sum them up. However, each time after one $A_{eligible}$ is relocated to a subprocess, the subprocesses whose pattern of data delivery is impacted need to update their respective $\{A_{eligible}\}$. Moreover, the possibility of an $A_{eligible}$ being selected and relocated by another subprocess at a later stage increases the uncertainty of identifying an optimal solution of reconstructing the business process. Therefore, to optimize temporal performance of one cluster is at least as hard as that of a single subprocess and can be perceived as an NP-hard problem as well. To efficiently approach the approximated optimal temporal performance of the overall business process, an array, namely $\{A_{abandon}\}$ is defined to store the $A_{eligible}$ that has been relocated to make sure that even $A_{eligible}$ might be optimal for other subprocesses at a later stage, they would not be relocated anymore. This process ensures the reduction of the number of the remaining $A_{eligible}$s.

My algorithm is initiated by detecting $\{A_{eligible}\}$ which is the set of the A$_{eligible}$s that are eligible to be relocated to other subprocesses. $\{A_{eligible}\}$ is then sorted along with $\{VA_{sum}\}$ in an ascending order. All the first elements of all the $\{VA_{sum}\}$ are then compared. The $A_{eligible}$ associated with the smallest VA_{sum} is relocated to its corresponding S_{target}. Then this $A_{eligible}$ is inserted to $\{A_{abandon}\}$, that indicates this $A_{eligible}$ cannot be relocated by any subprocess at a later stage. This process is iterated in the algorithm.

One S_{target} exits relocating $A_{eligible}$ when either its VA_{sum} associated

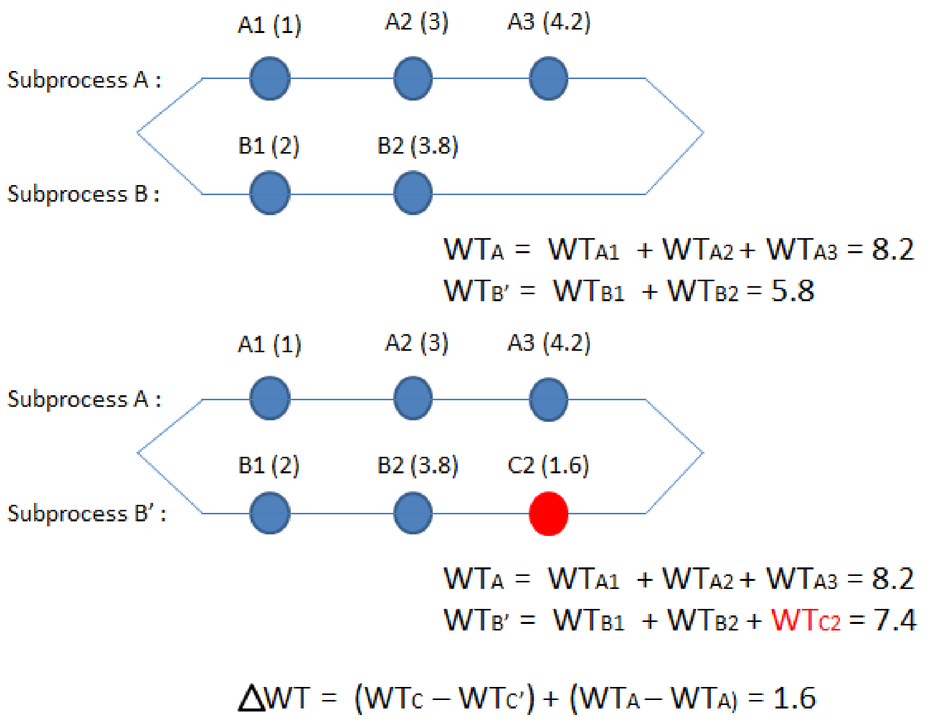
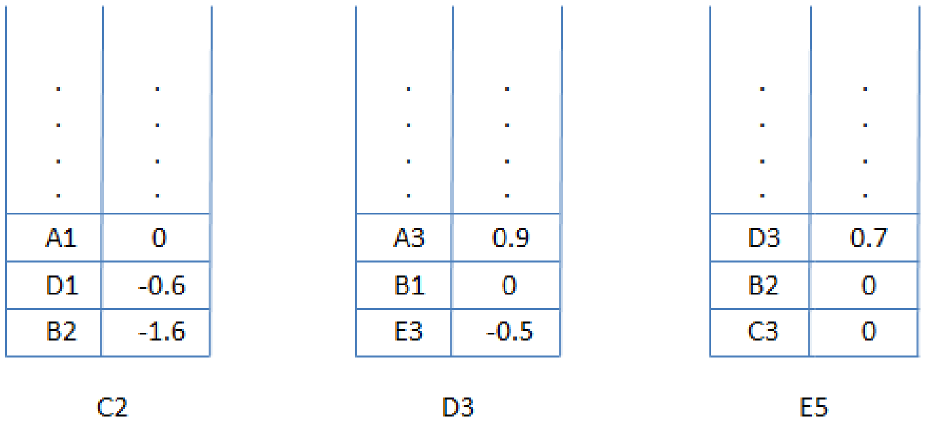


Figure 4.9: Updated working time of subprocess b after C2 being relocated

with the $A_{eligible}$ is a non-negative value, that means the overall time of executing business process would be increased after the relocation of the $A_{eligible}$, or no $A_{eligible}$ can be found.

The example of Figure 4.5 is used to illustrate the algorithm. C2, D3, and E5 are still assumed to be eligible to be relocated to subprocess B. Unlike the eligibly-relocated subprocess information that is maintained by each eligible activity as in Figure 4.5(b), each activity needs to maintain a table with the eligibly-relocated partial subprocess as the key and the working time gain as the value. The process to calculate working time gain is demonstrated in Figure 4.8 and Figure 4.9. After C2 is relocated from subprocess C, the working time gain of subprocess C is 1.6, as in Figure 4.8, while the working time gain of subprocess B after C2 being relocated is 0, as in Figure 4.9. Therefore, the overall working time gain by relocating C2 is calculated to be 1.6. The table of the eligibly-relocated subprocess and the corresponding working time gain can be reflected in Figure 4.10.



C2

.	.
.	.
.	.
.	.
A1	0
D1	-0.6
B2	-1.6

D3

.	.
.	.
.	.
.	.
A3	0.9
B1	0
E3	-0.5

E5

.	.
.	.
.	.
.	.
D3	0.7
B2	0
C3	0

Figure 4.10: The tables of eligibly-relocated subprocess and the corresponding working time gain maintained by eligible activities

All the values of each table then are collected and sorted in an ascending order, as in Figure 4.11, where the working time gains of C2, D3, and E5 are collected and sorted. The largest gain is obtained to retrieve the corresponding activity along with its eligibly-relocated subprocess. C2 is then identified and relocated.

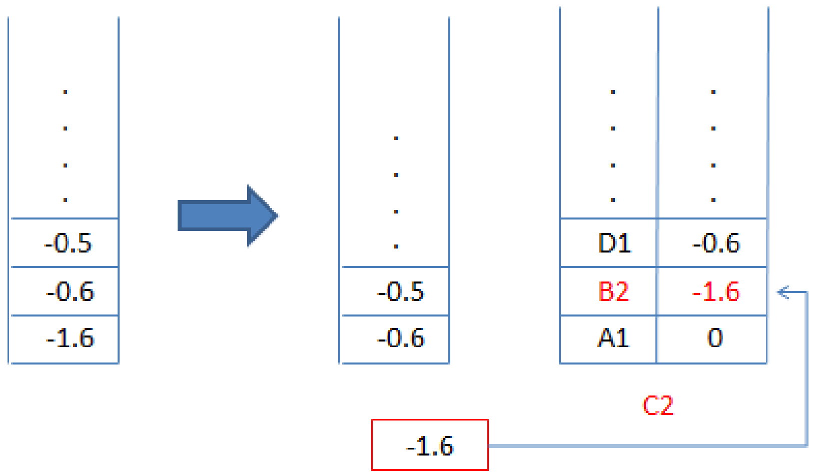


Figure 4.11: The process to sort the working time gains of all activities and select the corresponding eligibly-relocated subprocesses

Before proceeding to obtain the next largest working time gain of the sorted values, the activities and subprocesses associated with the relocated activity (C2) and its associated subprocesses (subprocess B and C) need to be updated, as in Figure 4.12, where D3 and E5 update their respective working time gain relevant to the subprocesses that C2 is originally located in (C) and relocated to (B). Then the above process is iterated until no working time gains can be obtained.

The pseudo code of this algorithm is listed in Algorithm 2. The process of calculating $VA_{original}$ and VA_{target} is omitted here for simplicity.

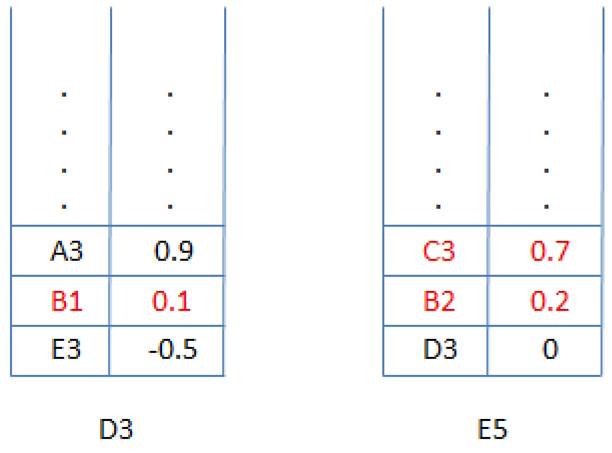


Figure 4.12: The updates of the rest activities and their respective working time gains with respect to the relocated activity and its associated subprocesses

Algorithm 2 Optimize_Time (Business_Process, Waiting_Time): deriving the optimized time for the overall business process

Input:
BP := business process that needs to optimize time performance,
WO_{BP} := working time of the overall business process, apriori knowledge
S_{target} := a cluster process that needs to optimize time performance,
$A_{eligible}$:= activity that is eligible for relocation to the cluster process that needs to optimize time performance,
$A_{abandon}$:= activity that has already been relocated to the cluster process that needs to optimize time performance, $\{A_{abandon}\} := \emptyset$,
$VA_{original}$:= varying working time of the original cluster where the eligible activity is located, apriori knowledge for simplicity
VA_{target} := varying working time of the target cluster where the eligible activity is located, apriori knowledge for simplicity,
VA_{sum} := varying working time of the business process contributed by the eligible activity after being relocated,
$VA_{interim}$:= interim element to represent the the smallest VA_{sum} of all the subprocesses,
$A_{interim}$:= interim element to represent the $A_{eligible}$ corresponding to its corresponding $VA_{interim}$,
Output:
Optimize_Time(BP, WO_{BP}) := the optimized working time of the overall business process

```
1: for each $S_{target} \in$ BP do
2:     $\{A_{eligible}\} := \emptyset$;
3:     $\{VA_{sum}\} := \emptyset$;
4:     $\{VA_{interim}\} := \emptyset$;
5:     $\{A_{interim}\} := \emptyset$;
6:     for each $A \notin (S_{target} \cup \{A_{abandon}\})$ in $BP$ do
7:         check whether $A$ is forwardly moveable, backwardly moveable, or loosely-parallelled
           moveable to $S_{target}$;
8:         if yes then
9:             $\{A_{eligible}\} := \{A_{eligible}\} + A$;
10:            $VA_{sum} := VA_{target} + VA_{original}$;
11:            $\{VA_{sum}\} := \{VA_{sum}\} + VA_{sum}$;
12:        end if
13:    end for
14:    sort($\{VA_{sum}\}$);
15:    sort($\{A_{eligible}\}$) according to sort($\{VA_{sum}\}$);
16:    $\{VA_{interim}\} := \{VA_{interim}\} + \{VA_{sum}\}[0]$;
17:    $\{A_{interim}\} := \{A_{interim}\} + \{A_{eligible}\}[0]$;
18: end for
19: sort($\{VA_{interim}\}$);
20: sort($\{A_{interim}\}$) according to sort($\{VA_{interim}\}$);
21: if $\{VA_{interim}\}[0] < 0$ and $\{A_{interim}\} \neq \emptyset$ then
22:     $WO_{BP} := WO_{BP} + \{VA_{sum}\}[0]$;
23:     $BP$ updates $\{A_{interim}\}[0]$;
24:     $\{A_{abandon}\} := \{A_{abandon}\} + \{A_{interim}\}[0]$;
25:     Optimize_Time($BP$, $WO_{BP}$);
26: else
27:     break;
28: end if
```

Chapter 5

Generating Symbolic Business Processes for Data-Centric Business Process Evaluation

This chapter is based on my paper entitled "Generating Symbolic Business Processes in Support of Evaluating Process Optimization" in the Proceedings of the 12th IEEE International Conference on Services Computing[1], coauthored by Professor Dewayne Perry, who supervised this paper and served as the correspondence author.

5.1 Summary

In this chapter, I introduce an approach, namely G-DCBP, to generate symbolic business processes, rather than depending on real-world business processes with their associated high cost, for evaluating approaches to optimize business processes.

[1]Yuqun Zhang and Dewayne E. Perry. Generating symbolic business processes in support of evaluating process optimization. In Services Computing (SCC), 2015 IEEE International Conference on, pages 754–758, June 2015.

5.2 Approach

G-DCBP maps the concepts of data-centric business processes to codes by: *1)* establishing the hierarchy that connects and layers *activity*[2], *subprocess*, *cluster*, and *process*; *2)* designing inputs in simple formats and outputting components relevant to data-centric business processes; *3)* symbolizing *data* in different data types, assigning them to *activities*, and forming *subprocesses* and *clusters*; *4)* defining data flow patterns among *activities*, *subprocesses*, and *clusters*; *5)* providing easy access for associating the symbolic business processes with the properties that need to be analyzed or evaluated.

5.2.1 Hierarchy

In G-DCBP, conceptual process components (i.e., activities, subprocesses, clusters, and processes) are correspondingly mapped to be the components: *activities*, *subprocesses*, *clusters*, and *processes*, that are implemented as classes in codes, where *subprocesses* inherit *activities*, *clusters* inherit *subprocesses*, and *processes* inherit *clusters*. That indicates that the derived components in the hierarchy can obtain the shared properties from their super components, from which their corresponding properties are aggregated or deduced. For instance, in the hierarchy of G-DCBP presented in Figure 5.1, by referencing data-setting methods (*setInitialData*, *setConsumedData*, etc.) of class *Activity*, class *Subprocess* distributes *data* to *activities* (though *assign-*

[2]In this chapter, italic expressions are used to represent components of G-DCBP, while the corresponding regular expressions are used to represent the theoretical concepts

DataToActivity, etc.). Class *Cluster* is equipped with methods to determine *cluster* types, and class *Process* comprehensively deploys all the instances of the other classes. Each of the classes is illustrated as follows:

5.2.1.1 Activity

An *activity* is the fundamental component of the hierarchy where the data types (e.g., *input_data,output_data*) and their associated settings are defined.

5.2.1.2 Subprocess

A *subprocess* assigns *data* to its associated *activities.* Micro-level data flows among *activities* within their associated *subprocess* are realized by *inherit_data_pool* that collects the *data* delivered from the executed *activities* and imputing them to the subsequently unexecuted *activities.*

5.2.1.3 Cluster

A *cluster* aggregates the corresponding *subprocesses*, where the macro-level data flows are realized by aggregating the subprocess-wise *inherit_data_pool* to deliver *data* to the subsequently unexecuted *clusters.*

A *cluster* can be designed as *beginning_cluster*, *join_cluster*, *split_cluster*, *join_and_split_cluster*, or *singular_cluster*, due to their corresponding theoretical concepts illustrated in Chapter 3.

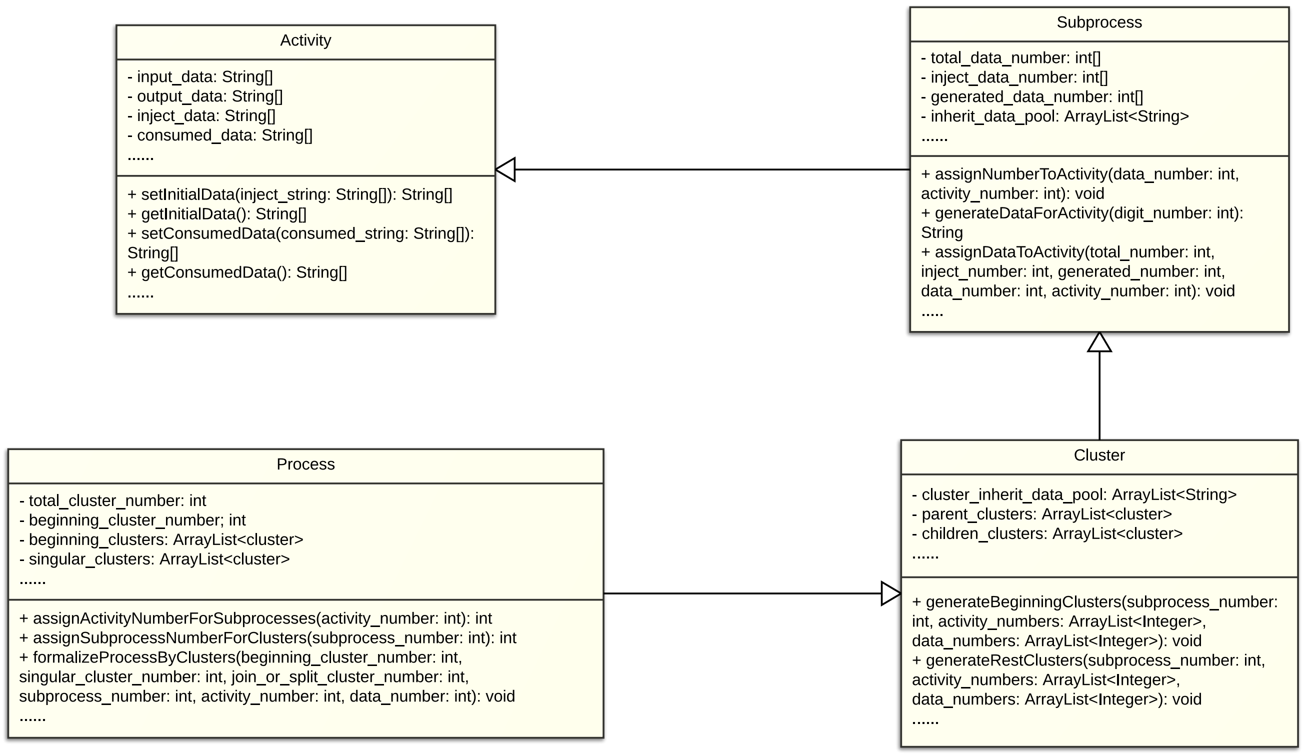

Activity
- input_data: String[]
- output_data: String[]
- inject_data: String[]
- consumed_data: String[]
......
+ setInitialData(inject_string: String[]): String[]
+ getInitialData(): String[]
+ setConsumedData(consumed_string: String[]): String[]
+ getConsumedData(): String[]
......
Subprocess
- total_data_number: int[]
- inject_data_number: int[]
- generated_data_number: int[]
- inherit_data_pool: ArrayList<String>
......
+ assignNumberToActivity(data_number: int, activity_number: int): void
+ generateDataForActivity(digit_number: int): String
+ assignDataToActivity(total_number: int, inject_number: int, generated_number: int, data_number: int, activity_number: int): void
.....
Process
- total_cluster_number: int
- beginning_cluster_number; int
- beginning_clusters: ArrayList<cluster>
- singular_clusters: ArrayList<cluster>
......
+ assignActivityNumberForSubprocesses(activity_number: int): int
+ assignSubprocessNumberForClusters(subprocess_number: int): int
+ formalizeProcessByClusters(beginning_cluster_number: int, singular_cluster_number: int, join_or_split_cluster_number: int, subprocess_number: int, activity_number: int, data_number: int): void
......
Cluster
- cluster_inherit_data_pool: ArrayList<String>
- parent_clusters: ArrayList<cluster>
- children_clusters: ArrayList<cluster>
......
+ generateBeginningClusters(subprocess_number: int, activity_numbers: ArrayList<Integer>, data_numbers: ArrayList<Integer>): void
+ generateRestClusters(subprocess_number: int, activity_numbers: ArrayList<Integer>, data_numbers: ArrayList<Integer>): void
......

Figure 5.1: The G-DCBP class Diagram

5.2.1.4 Process

A process is composed of *clusters* and configured by their placements. *Process* applies stochastic models (e.g., uniform, Gaussian) to determine the runtime values of the corresponding input parameters for *activities*, *subprocesses*, and *clusters*. Typically, *subprocesses* are generated one by one with their associated *activities* and *data* assigned. *Clusters* are subsequently generated by aggregating their associated *subprocesses* one by one as well. Lastly, *clusters* are placed according to their types, and a symbolic business process is established.

5.2.2 System I/O

G-DCBP aims at offering end users easy access by generating symbolic data-centric business processes according to their scenario settings of process components. The detailed input parameters are listed as follows:

- *total_cluster_number*: the total number of the *clusters* that are needed in the expected business processes
- *beginning_cluster_number*: the number of the *clusters* that indicate the beginning of the execution of business processes
- *singular_cluster_number*: the number of the *clusters* whose execution is triggered by and triggers one *cluster*
- *connecting_cluster_number*: the total number of the *join_cluster*, *split_cluster*, *join_and_split_cluster*

- *mean_number_of_subprocesses_in_cluster*: the mean number of *subprocesses* in one *cluster* from which the runtime number of *subprocesses* in each *cluster*, namely *number_of_subprocesses_in_cluster* is derived in accordance with the stochastic models (e.g., uniform, Gaussian).

- *mean_number_of_activities_in_subprocess*: the mean number of *activities* in one *subprocess* from which the runtime number of *activities* in each *subprocess*, namely *number_of_activities_in_subprocess* is derived in accordance with the stochastic models.

- *mean_data_dependency_rate*: the mean value of *data_dependency_rate* that is defined as the *data* assigned to one *subprocess* from the associated *cluster* divided by the total number of the *data* in each associated cluster; the runtime *data_dependency_rate* is derived by this mean and the stochastic models.

- *mean_number_of_data_per_cluster*: the mean number of *data* in one *cluster* from which the runtime number of data in each *cluster*, namely *number_of_data_in_cluster* is derived in accordance with the stochastic models.

Given the deterministic input parameters regarding *clusters*, the mean input parameters of *data*, *activities*, and *subprocesses*, and the pre-determined s-tochastic models, G-DCBP generates data-centric business processes and all the relevant components that meet end users' expectations under such simple input formats.

5.2.3 Symbolization

In G-DCBP, data are symbolized as the combination of English letters, e.g., “AE”, “BYCS”, with its digits determined by *number_of_data_per_cluster*, *data_dependency_rate*, and the *data* that have already been designed. Particularly, *data* is generated per *cluster* and assigned to the underlying *subprocesses*.

The digit number of data symbolization can be calculated as equation (5.1), where n denotes *number_of_data_per_cluster* of the associated *cluster*, D_i denotes the number of *data* in the ith *cluster*, and m denotes the number of the *clusters* that have been generated. An example is used to illustrate the mechanism. Assume 100 pieces of *data* are generated for a *cluster* c. The number of the assigned *data* for the generated *clusters* is 2000. It can be derived that this scenario can be applied with the third sub-equation of equation (5.1). Therefore the digit number of the data symbolization in c is $ceil(log_{26}2000)$, which is 3.

$$
digit\_number = \begin{cases}
ceil(\log_{26} n), & 26^{ceil(\log_{26} n)} - \|\sum_{i=0}^{m} D_i\| \geq n \\
ceil(\log_{26} n)+1, & 0 \leq 26^{ceil(\log_{26} n)} - \|\sum_{i=0}^{m} D_i\| \\
 & < n \\
ceil(\log_{26} \|\sum_{i=0}^{m} D_i\|), & 26^{ceil(\log_{26} n)} < \|\sum_{i=0}^{m} D_i\|, \\
 & ceil(\log_{26} \|\sum_{i=0}^{m} D_i\|) = \\
 & ceil(\log_{26}(\|\sum_{i=0}^{m} D_i\| + \mathrm{n})) \\
ceil(\log_{26}(\|\sum_{i=0}^{m} D_i\| + \mathrm{n})), & 26^{ceil(\log_{26} n)} < \|\sum_{i=0}^{m} D_i\|, \\
 & ceil(\log_{26} \|\sum_{i=0}^{m} D_i\|) < \\
 & ceil(\log_{26}(\|\sum_{i=0}^{m} D_i\| + \mathrm{n}))
\end{cases}
\tag{5.1}
$$

During the phase of generating one *subprocess* s, the number of the *data* assigned from the generated data of the associated *cluster c*, namely $number\_of\_data_s$ is calculated as $number\_of\_data\_per\_cluster_c$ * $data\_dependency\_rate_s$. The corresponding *data* are randomly determined from the generated *data* in *c* afterwards. The generation of these *data* follows the alphabetic order.

The entire process above is iterated to assign *data* for every *subprocess.*

5.2.4 Flow Patterns

Data-centric business processes depict the overall effects of business processes by data flows that are represented as the transitions and transmissions between data types, in both macro- and micro-levels.

To establish micro-level data flows, G-DCBP generates *data* according to the input parameters, assign them to each *activity*, and define them as different data types (e.g., *input_data, consumed_data*). This completion of the assignments of the data types for each *activity* reflects the execution order of *activities*, and therefore establishes micro-level data flows.

Typically, a piece of data can be consumed by the execution of its activity, delivered to the subsequent activities for their executions, or ended being unused after executing its associated activity. However, flow patterns can be complicated, for instance, when data is not instantaneously delivered to the subsequently unexecuted activities for their executions. In G-DCBP, these *data* are collected by *Inherit_data_pool*, that is applied both subprocess-wise and cluster-wise. A subprocess-wise *Inherit_data_pool* collects the *input_data* and the *inject_data* that has not been consumed, and the *output_data* each time after completing the design of an *activity*. For each *activity*, its *input_data* is determined from its associated *Inherit_data_pool*. This process of updating *Inherit_data_pool* and selecting the *input_data* is iterated for every *activity* to enable the micro-level data flows. Similar to subprocess-wise *Inherit_data_pool*s, cluster-wise *Inherit_data_pool*s collect subprocess-wise *Inherit_data_pool*s within the same *clusters*.

Macro-level data flows can be illustrated by examples. Assume 2 activities, labeled 1 and 2, and 5 pieces of data, labeled A to E, are assigned to a subprocess. In an G-DCBP implementation, activity 1 is assigned with data A, B, C, where A and B are the *initial_data*, C is the *output_data*, and A is consumed by the execution. The subprocess-wise *Inherit_data_pool* collects B and C as the possible *input_data* delivered to activity 2, that takes B as its *input_data* and D as its *initial_data*, B then is consumed and E is its *output_data*. The instantaneous subprocess-wise *Inherit_data_pool* then is updated with C, D and E.

Macro-level data flows are realized by cluster-wise *Inherit_data_pool*s, that are formed by aggregating *data* from its associated subprocess-wise *Inherit_data_pool*s bounded by convergence/divergence relations. Moreover, since the neighboring *clusters* that are executed before (known as *parent_cluster*) and after (known as *child_cluster*) the current *cluster* is identified in the *cluster* component of G-DCBP, inter-cluster data delivery can be realized by simply acquiring the *Inherit_data_pool* form the *parent_cluster* and delivering its own *Inherit_data_pool* to the *child_cluster*, thus ensuring the macro-level data flows.

5.2.5 Easy Access

G-DCBP delivers business processes with straightforward hierarchy of concepts, i.e., *activities*, *subprocesses*, and *clusters*, that can be easily accessed by simply referencing their classes and creating the corresponding objects, fields, and methods. Moreover, the *process* module collects all the *activities, subprocesses,* and *clusters* for easy access by end users.

5.3 A Case Study of the Usability of G-DCBP

In this section, G-DCBP is used to evaluate the efficacy of an approach to improve the temporal performance of data-centric business processes that is proposed in my previous work [67]. My approach in [67] improves the temporal performance of data-centric business processes by relocating the placement of business activities in the business processes and thus changing the execution order of business activities such that the overall working time can be reduced.

The approach includes two parts: eligibility verification for business activity relocation and the algorithm to change the execution order of business activities, both of which are mapped and implemented in G-DCBP.

5.3.1 Eligibility Verification for Business Activity Relocation

Given a subprocess and a business activity, the execution order of the business activity can be modified when that activity can be **forwardly moveable**, **backwardly moveable**, or **loosely-parallelled moveable** to that subprocess, as illustrated in [67].

For each *activity* that is created in G-DCBP, it is designed with an instantaneous *Inherit_data_pool*, that is updated by the data types of this *activity*. When a data-centric business process is generated by G-DCBP with its components (*clusters*, *subprocesses*, *activities*, etc) being delivered, each *activity* is validated for which of the partial subprocess (*activities*) it can be relocated to by applying the rules of **forwardly moveable**, **backwardly moveable**, and **loosely-parallelled moveable**. All the partial subprocess (*activities*) where an *activity* can be **forwardly moveable**, **backwardly moveable**, or **loosely-parallelled moveable** are collected.

5.3.2 Algorithms to Improve Temporal Performances

Assuming each activity is assigned with its working time (derived or estimated) and has been verified whether it is eligible to be relocated to other subprocesses, the algorithm to improve temporal performance of data-centric business processes in [67] collects the eligible business activities, sorts their respective temporal gains for the subprocesses that they can be relocated to, and relocates the activities to their corresponding subprocesses according to their rankings for an optimal overall temporal performance of the business processes.

5.3.2.1 Time Assignments

In G-DCBP implementations, working time is randomly generated and assigned to each *activity*. Therefore, the working time of *subprocesses* can be derived by simply summing the working time of their associated *activities*, and the working time of *clusters* is calculated as the maximum working time of their associated *subprocesses*.

5.3.2.2 Temporal Gain Calculations and Rankings

In this chapter, the optimized business goal is the average working time of clusters. Therefore, the temporal gain of moving an *activity* is defined as the difference between the temporal gain of its originally associated *cluster* after being moved from, namely G_{cf} and the temporal gain of its newly associated *cluster* after being moved to, namely G_{ct}. Specifically, the temporal gains for moving an *activity* are calculated as follows:

$$temporal\_gain = G_{cf} - G_{ct} \tag{5.2}$$

G_{cf} and G_{ct} calculated as equation (5.3) and (5.4), where WT_A denotes the working time of the *activity* A, WT_{sf} denotes the working time of A's originally associated *subprocess* where A is moved from, WT_{cf} denotes the working time of A's originally associated *cluster* where A is moved from, $max\{\mathrm{WT}_{si}\}$ denotes the updated maximum working time among all the *subprocesses* within the *cluster* where A is originally associated with after A being moved out, WT_{st} denotes the working time of A's newly associated *subprocess* where A is moved to, and WT_{ct} denotes the working time of A's newly associated *cluster* where A is moved to.

$$G_{cf} = \begin{cases} WT_A, & WT_{sf} = WT_{cf}\ and\ WT_{sf} - WT_A \\ & = max\{WT_{si}\} \\ WT_{cf} - max\{WT_{si}\}, & WT_{sf} = WT_{cf}\ and\ WT_{sf} - WT_A \\ & < max\{WT_{si}\} \\ 0, & WT_{sf} < WT_{cf} \end{cases} \tag{5.3}$$

$$G_{ct} = \begin{cases} WT_A, & WT_{st} = WT_{ct} \\ WT_{st} + WT_A - WT_{ct}, & WT_{st} < WT_{ct}\ and\ WT_{st} + WT_A > WT_{ct} \\ 0, & WT_{st} < WT_{ct}\ and\ WT_{st} + WT_A <= WT_{ct} \end{cases} \tag{5.4}$$

To record the partial *subprocesses* (*activities*) that each *activity* can be moved to and the corresponding temporal gains, a lookup table, namely *activity_gain_table* is designed for each *activity*, with the partial *subprocesses* (*activities*) as keys and the corresponding gains as values. Furthermore, all the temporal gains from all the *activities* are sorted in a list, namely *activity_gain_rankings* in descending order.

5.3.3 Implementation of Business Activity Relocation

The temporal performance improvements start off by obtaining the top element in *activity_gain_rankings* and comparing it with each *activity*'s *activity_gain_table*. After matching the value of the entry in the *activity_gain_table*, the associated *activity* A can be found and the corresponding partial *subprocess* S can be identified as the partial *subprocess* that is expected to improve the temporal performance of the business processes after A being relocated to it. A is then relocated to S with the relevant information (e.g., *activity_gain_rankings*) being updated.

Subsequently, A is marked as *unmoveable* that indicates it would not be relocated to other *subprocesses* even when temporal performance of business processes can be improved, as explained in [67]. A's corresponding entry in *activity_gain_table* is deleted accordingly. This process repeats until all the eligible *activities* are iterated and relocated to their respective *subprocesses* before an optimal temporal performance of business processes is approached.

5.3.4 Evaluations

5.3.4.1 Scenarios

Two basic scenarios are adopted to evaluate my approach to improve the temporal performance of data-centric business processes, with the uniform distribution:

- light-load scenario:
  - *total_cluster_number*: 10
  - *beginning_cluster_number*: 2
  - *singular_cluster_number*: 6
  - *connecting_cluster_number*: 2
  - *mean_number_of_subprocesses_in_cluster*: 2
  - *mean_number_of_activities_in_subprocess*: 5
  - *mean_data_dependency_rate*: 100%
  - *mean_number_of_data_in_subprocess*: 30

- heavy-load scenario:
  - *total_cluster_number*: 100
  - *beginning_cluster_number*: 10
  - *singular_cluster_number*: 80
  - *connecting_cluster_number*: 10
  - *mean_number_of_subprocesses_in_cluster*: 5
  - *mean_number_of_activities_in_subprocess*: 10
  - *mean_data_dependency_rate*: 100%
  - *mean_number_of_data_in_subprocess*: 60

All of the simulation scenarios in this thesis are adjusted in terms of some variables of these two basic scenarios.

5.3.4.2 Metrics

- *temporal improvement ratio*: calculated as the increment ratio of the average working time of clusters after being applied with the temporal improvement algorithm divided by the average working time of clusters before being applied with the temporal improvement algorithm

- *occurrence ratio*: defined as the number of runs where the improvement of the temporal performance of business process occur divided by the number of all runs in G-DCBP.

5.3.4.3 Results

To better understand the causality of the metrics and the process-relevant factors, the evaluation is conducted with multiple groups of *mean_number_of_activities_in_subprocess* (i.e., 5, 10, and 15 average activities per subprocess).

light-load scenario in terms of mean activity working time Figure 5.2 shows the temporal improvement ratio in terms of the mean activity working

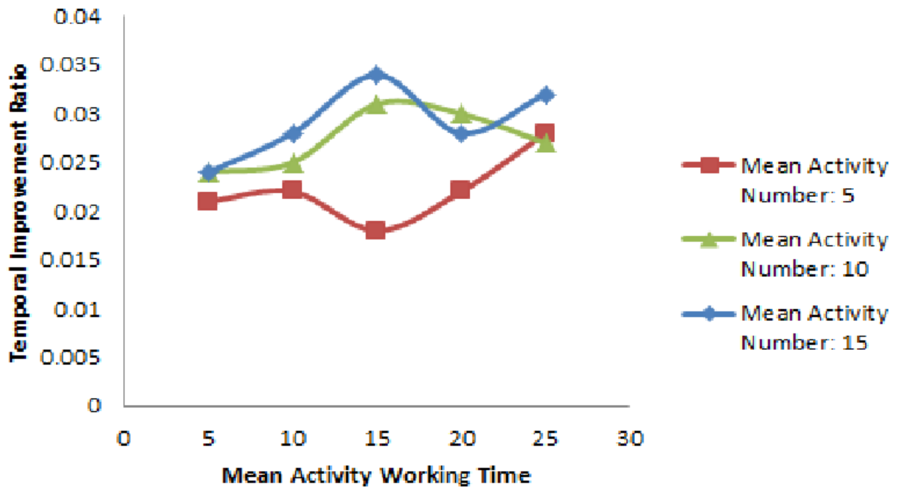


Figure 5.2: temporal improvement ratio vs. mean activity working time in light-loaded scenario

time ranging from 5 to 25 of the light-load scenarios. It can be observed that the temporal improvement ratio does not vary significantly with the updates of the mean activity working time (in general between 1.5% and 3.5%).

What is more, though the subprocesses with more average activity number generally have better temporal gains, they are not significant.

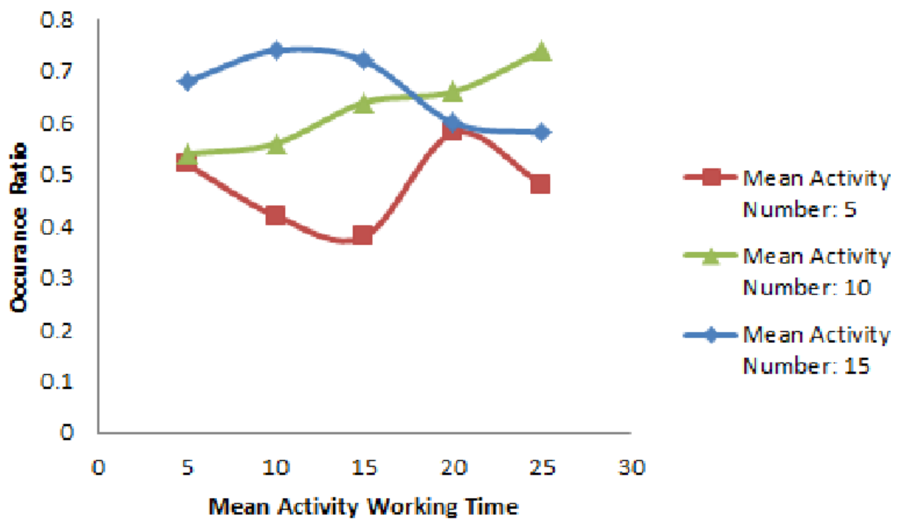


Figure 5.3: occurrence ratio vs. mean activity working time in light-loaded scenario

Figure 5.3 illustrates the occurrence ratio in terms of the varying mean activity working time (in general 37% to 75%). Similar to the temporal improvement ratio performance, the occurrence ratio of temporal improvement is not significantly correlated with the mean activity working time and *mean_number_of_activities_in_subprocess*.

light-load scenario in terms of *mean_number_of_data_per_cluster* Figure 5.4 illustrates the temporal improvement ratio in terms of *mean_number_of-_data_per_cluster*. It can be observed that temporal improvement ratio is not significantly correlated with *mean_number_of_data_per_cluster*, while it is more significant that the subprocesses with fewer activities tend to have better temporal gains than the ones with more when they have more data for executions.

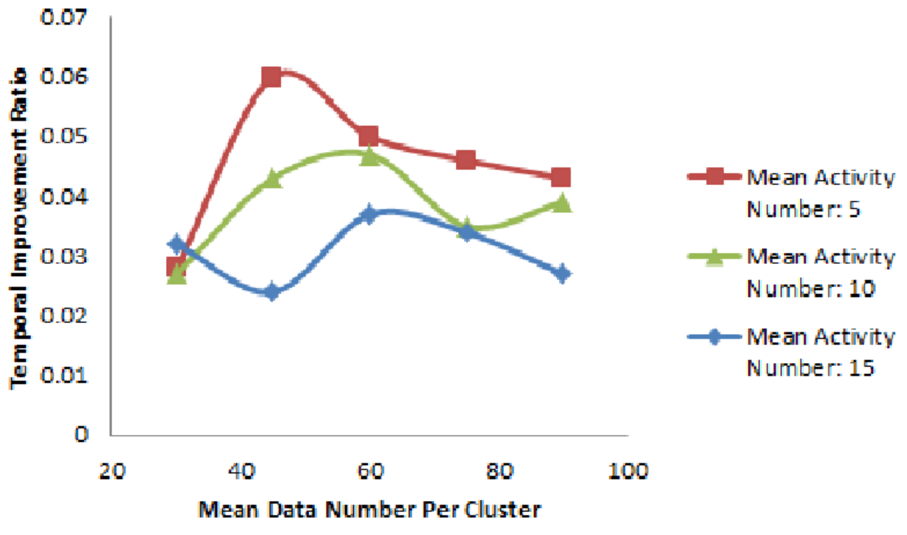


Figure 5.4: temporal improvement ratio vs. *mean_number_of_data_in_subprocess* in light-loaded scenario

Figure 5.5 illustrates the occurrence ratio in terms of *mean_number_of_d-ata_per_cluster*. It can be observed that the occurrence ratio is not significantly correlated with either *mean_number_of_activities_in_subprocess* or *mean_numb-er_of_data_per_cluster*.

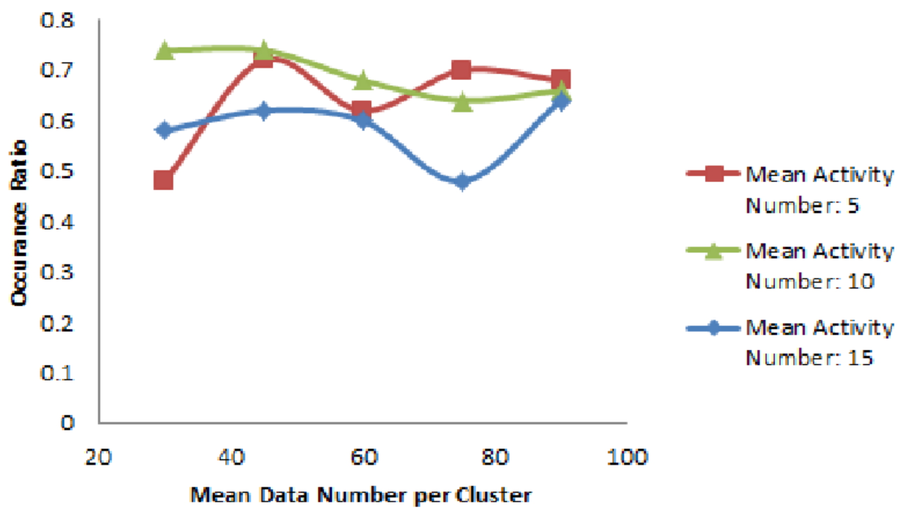


Figure 5.5: occurrence ratio vs. *mean_number_of_data_in_subprocess* in light-loaded scenario

It can be speculated that under light-loaded scenarios, the temporal performance of business processes does not have much to do with subprocess-wise properties, e.g., *mean_number_of_data_per_cluster*, etc.

heavy-load scenario in terms of *mean_activity_working_time* Figure 5.6 shows the temporal improvement ratio in terms of mean activity working time in the heavy-load scenarios. Similar to the light-load scenario, it can be observed that the temporal improvement ratio is not significantly correlated with either mean activity working time or *mean_number_of_activities_in_subprocess*. On the other hand, compared with the temporal improvement ratio in the light-load scenarios, the temporal improvement ratio in heavy-load scenario is significantly higher (15% to 23% compared with 1.5% to 3.5%).

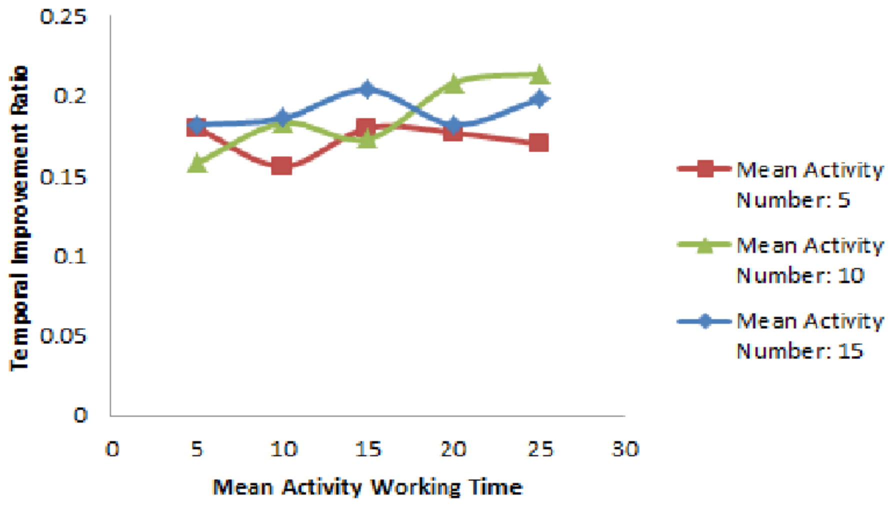


Figure 5.6: temporal improvement ratio vs. mean activity working time in heavy-loaded scenario

Figure 5.7 depicts the occurrence ratio in terms of mean activity working time in the heavy-load scenarios. Obviously, under the settings of the heavy-load scenario, the temporal improvement occurs in every configuration of data-centric business processes.

heavy-load scenario in terms of *mean_number_of_data_per_cluster*

Figure 5.8 illustrates the temporal improvement ratio in terms of *mean_numbe-r_of_data_per_cluster* in the heavy-load scenarios. It can be observed that the temporal improvement ratio does not vary by too much among different *mean_number_of_activities_in_subprocess*. Moreover, the results imply a weak

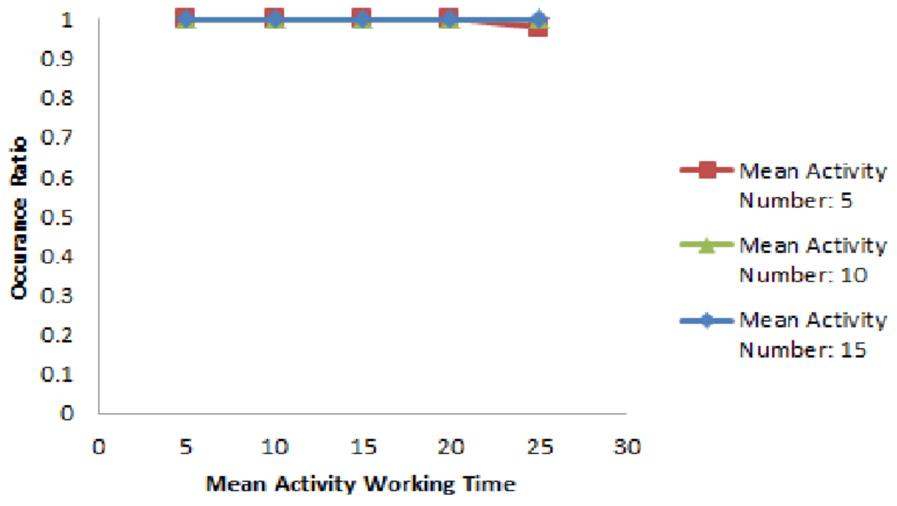


Figure 5.7: occurrence ratio vs. mean activity working time in heavy-loaded scenario

pattern that by increasing *mean_number_of_data_per_cluster*, the temporal improvement ratio tends to be lower.

Figure 5.9 indicates that under the settings of the heavy-load scenario, the temporal improvement occurs in every configuration of data-centric business processes, no matter how may data items are distributed to each subprocess.

Compared with the temporal performances in the light-load scenario, the temporal performance in the heavy-load scenario are better in terms of both temporal improvement ratio and occurrence ratio.

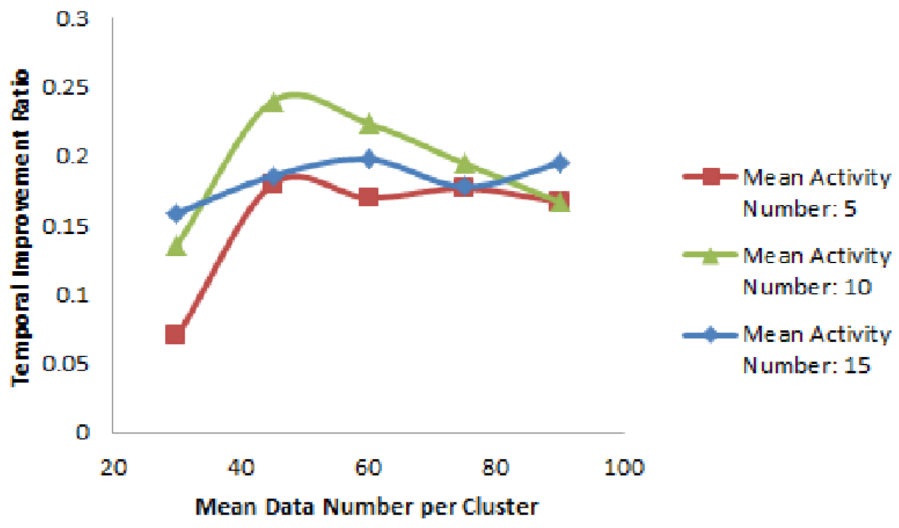


Figure 5.8: temporal improvement ratio vs. *mean_number_of_data_in_subprocess* in heavy-loaded scenario

heavy-load scenario of 50% *mean_data_dependency_rate* In real world, processes with high data dependency can be limited, e.g., representing repeated business services for different customers. In this section, I investigate the affect on the temporal improvement from data dependency between subprocesses. Specifically, the *mean_data_dependency_rate* is set to be 50%.

Figure 5.10 illustrates the temporal improvement ratio in terms of *mean_number_of_data_in_subprocess*. Compared with the heavy-load scenario with 100% *mean_data_dependency_rate*, the temporal improvement ratio of the heavy-load scenario with 50% is significantly weaker. For instance, for the processes with 10 mean activity number, the temporal improvement ratio of the heavy-load scenario with 100% *mean_data_dependency_rate* is around

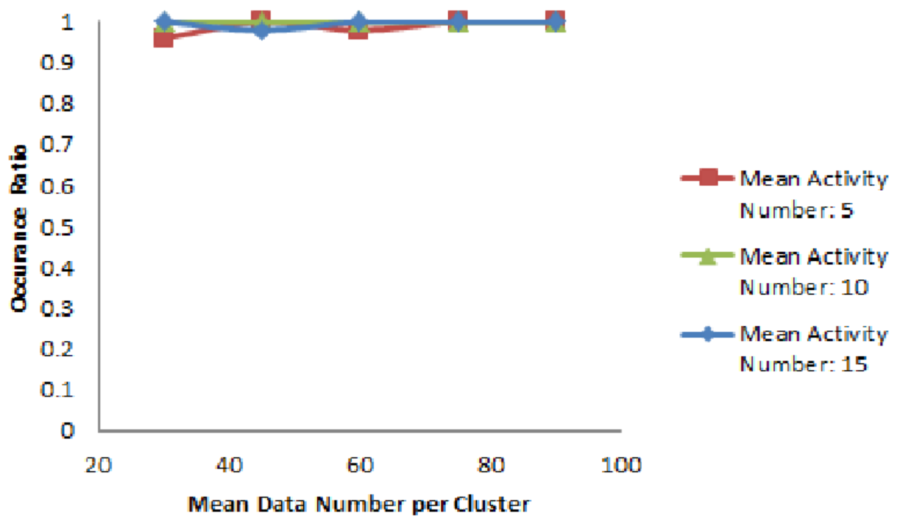


Figure 5.9: occurrence ratio vs. *mean_number_of_data_in_subprocess* in heavy-loaded scenario

13% to 25%, while it is only around 1% to 4% in the heavy-load scenario with 50% *mean_data_dependency_rate*. In addition, it also can be observed that the processes with more *mean_number_of_activities_in_subprocess* tend to have better temporal improvement ratio performance (0.1% to 1% of 5 *mean_number_of_activities_in_subprocess* and 1% to 4% of 10 *mean_number_of_a-ctivities_in_subprocess*). On the other hand, by increasing *mean_number_of_dat-a_per_cluster*, the temporal improvement ratio significantly decreases when *mean_data_dependency_rate* is 50%.

Figure 5.11 indicates the occurrence ratio in terms of *mean_number_of_d-ata_per_cluster*. It can be observed that the processes with fewer *mean_number-_of_activities_in_subprocess* (e.g., 5) has a lower occurrence ratio than the pro-

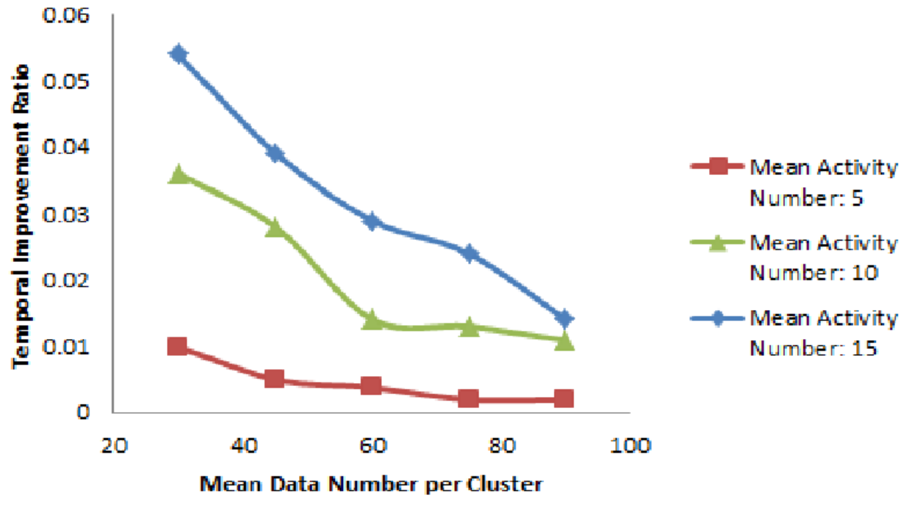


Figure 5.10: temporal improvement ratio vs. *mean_number_of_data_in_subprocess* in heavy-loaded scenario of 50% *mean_data_dependency_rate*

cesses with more.

5.3.4.4 Analysis of Evaluation Results

The simulation using G-DCBP is established under the uniform distribution model, which implies that the analysis below might not be fit for the evaluation that is based on other types of stochastic models.

- The performance of temporal optimization of data-centric business processes mainly relies on how subprocesses are involved with each other with respect to their data dependencies. The higher data dependencies are, the more chances business activities can be relocated and the better

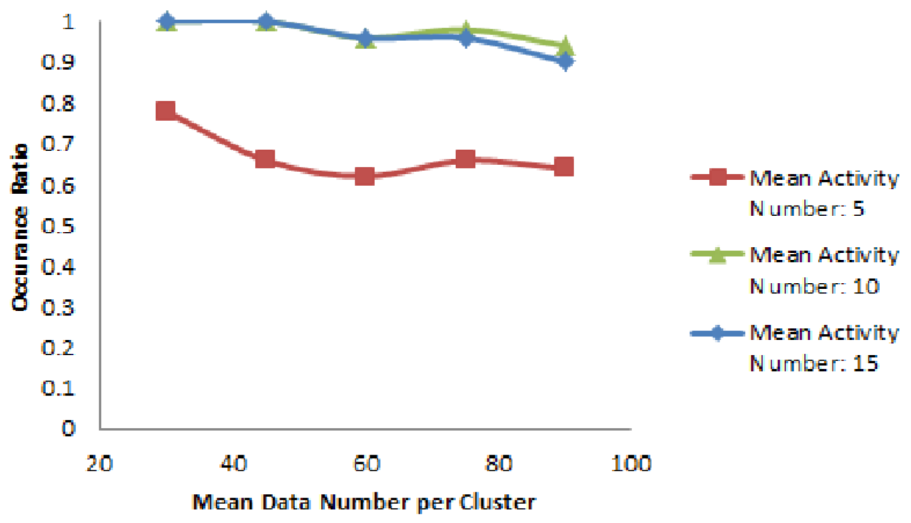


Figure 5.11: occurrence ratio vs. *mean_number_of_data_in_subprocess* in heavy-loaded scenario of 50% *mean_data_dependency_rate*

the temporal improvement can be.

- Under the same level of data dependencies, the temporal improvement performance is mainly subject to the size of the process. Typically, large-size data-centric business process tends to have better temporal improvement performance than the small-size ones.

- Besides data dependencies and the size of business processes, the number of data items of a subprocess can affect the temporal improvement of data-centric business processes. This affect is more significant for large-size business processes. Typically, in large-size data-centric business processes, the temporal improvement ratio is lower when the number of data items of each subprocess is larger.

- The parameters relevant to activities (activity working time, activity number per subprocess) do not impact significantly on the temporal improvement performance of data-centric business processes.

Chapter 6

Predict Inventory Shipments

This chapter is based on my paper entitled "Predict Inventory Shipments of Oracle EBS Systems" in the Proceedings of the 13th IEEE International Conference on Services Computing[1], coauthored by Professor Dewayne Perry, who supervised this paper and served as the correspondence author.

6.1 Summary

In this chapter, I present a framework to forecast typical business process data, (e.g., inventory shipments) and have it evaluated with real-world data sets extracted from Oracle EBS systems.

6.2 Extracting Inventory Transactions

The inventory management process of the company *A* includes purchasing and assembling parts, shipping them to customers, etc., all of which occur in its warehouses. Particularly, these warehouses are designed with respective functions: storing purchased parts and having them ready for assemblies,

[1]Yuqun Zhang and Dewayne E. Perry. Predict Inventory Shipments of Oracle EBS Systems. In Services Computing (SCC), 2016 IEEE International Conference on, June 2016.

storing assembled products and having them ready to be transferred to the shipment warehouses, shipping products out to customers, storing returned products, etc. For each function, there are one or multiple physical warehouses that are made for certain parts or products.

In A's Oracle EBS system, these warehouses are conceptually mapped as "Inventory Organization", and each of them is defined as a "Subinventory". Specifically, their codes and functions are listed in Table 6.1.

Subinventory code	function
BULK	Store returned items and certain purchased parts
GUADFGI	Store products after assembly
INTRAN	Store certain purchased parts and products and distribute them to other Subinventories
MRB	Store certain products and purchased parts
GUADPACK	Assemble parts to be products
GUADSTAG	Store products that are ready to be shipped out

Table 6.1: Subinventory lists

A typical inventory management process of A can be as follows. After a purchase order is made, some parts are purchased and stored in INTRAN. They are transferred later to GUADPACK for assembly in workshops according to certain manufacture orders. Then the products are placed in GUADFGI after assembly in workshops. When a sales order is made, products are transferred to GUADSTAG for being shipped out to customers.

The data set of our study is the inventory transactions that occurred within one year, that are generated from the inventory management module of

Oracle EBS system. Its .csv form is illustrated as Figure 6.1, where each entry is an inventory transaction. For instance, entry 3 indicates that at $8:55am$, November 25, 2015, 2 pieces of the item with the code 88014-00 are shipped out (transaction_type is “sales order issue”) from GUADSTAG. Moreover, entry 5 indicates that at $7:13am$, October 2, 2015, 10 items with the code 92000-00 are transferred from GUADFGI to GUADSTAG because of a sales order (transaction_type is “sales order pick”). There are 32162 entries in our data set.

Since our goal in the chapter is to predict inventory shipments for products, the target Subinventory code is set to be GUADSTAG, and 10 products with the most shipment frequencies are selected for studies.

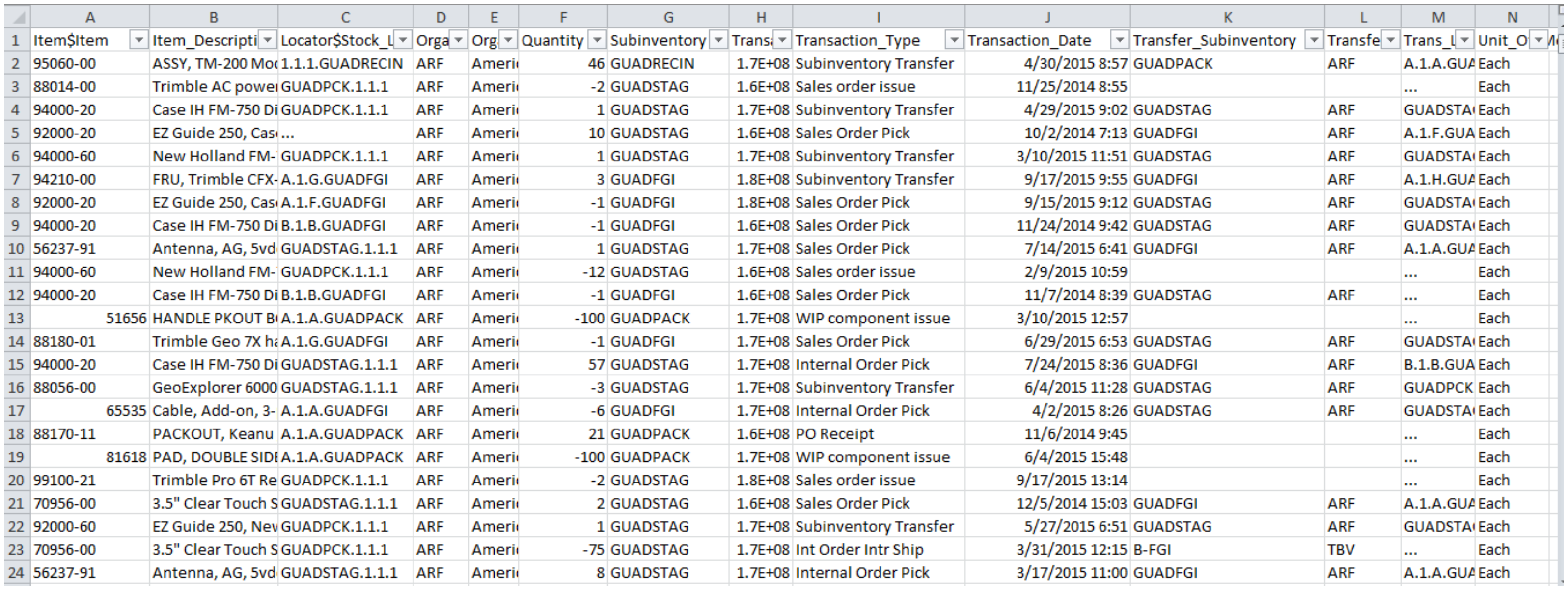
6.3 Predicting Inventory Shipments

In this section, we present out methodologies of predicting shipments. Existing approaches, i.e., ARMA and the Primitive KNN, are first used for shipment predictions. Then we propose new algorithms based on KNN algorithms that can show that our approach is at least as accurate and effective and requires significantly less computation time.

6.3.1 The Methodology with ARMA

6.3.1.1 ARMA Algorithm

Autoregressive moving average (ARMA) model is proposed for understanding and predicting future values by a given time series of data [70]. This



	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	Item$Item	Item_Descripti	Locator$Stock_L	Orga	Org	Quantity	Subinventory	Trans	Transaction_Type	Transaction_Date	Transfer_Subinventory	Transfe	Trans_L	Unit_O
2	95060-00	ASSY, TM-200 Mo	1.1.1.GUADRECIN	ARF	Ameri	46	GUADRECIN	1.7E+08	Subinventory Transfer	4/30/2015 8:57	GUADPACK	ARF	A.1.A.GUA	Each
3	88014-00	Trimble AC powe	GUADPCK.1.1.1	ARF	Ameri	-2	GUADSTAG	1.6E+08	Sales order issue	11/25/2014 8:55			...	Each
4	94000-20	Case IH FM-750 Di	GUADPCK.1.1.1	ARF	Ameri	1	GUADSTAG	1.7E+08	Subinventory Transfer	4/29/2015 9:02	GUADSTAG	ARF	GUADSTA	Each
5	92000-20	EZ Guide 250, Cas	...	ARF	Ameri	10	GUADSTAG	1.6E+08	Sales Order Pick	10/2/2014 7:13	GUADFGI	ARF	A.1.F.GUA	Each
6	94000-60	New Holland FM-	GUADPCK.1.1.1	ARF	Ameri	1	GUADSTAG	1.7E+08	Subinventory Transfer	3/10/2015 11:51	GUADSTAG	ARF	GUADSTA	Each
7	94210-00	FRU, Trimble CFX-	A.1.G.GUADFGI	ARF	Ameri	3	GUADFGI	1.8E+08	Subinventory Transfer	9/17/2015 9:55	GUADFGI	ARF	A.1.H.GUA	Each
8	92000-20	EZ Guide 250, Cas	A.1.F.GUADFGI	ARF	Ameri	-1	GUADFGI	1.8E+08	Sales Order Pick	9/15/2015 9:12	GUADSTAG	ARF	GUADSTA	Each
9	94000-20	Case IH FM-750 Di	B.1.B.GUADFGI	ARF	Ameri	-1	GUADFGI	1.6E+08	Sales Order Pick	11/24/2014 9:42	GUADSTAG	ARF	GUADSTA	Each
10	56237-91	Antenna, AG, 5vd	GUADSTAG.1.1.1	ARF	Ameri	1	GUADSTAG	1.7E+08	Sales Order Pick	7/14/2015 6:41	GUADFGI	ARF	A.1.A.GUA	Each
11	94000-60	New Holland FM-	GUADPCK.1.1.1	ARF	Ameri	-12	GUADSTAG	1.6E+08	Sales order issue	2/9/2015 10:59			...	Each
12	94000-20	Case IH FM-750 Di	B.1.B.GUADFGI	ARF	Ameri	-1	GUADFGI	1.6E+08	Sales Order Pick	11/7/2014 8:39	GUADSTAG	ARF	...	Each
13	51656	HANDLE PKOUT B	A.1.A.GUADPACK	ARF	Ameri	-100	GUADPACK	1.7E+08	WIP component issue	3/10/2015 12:57			...	Each
14	88180-01	Trimble Geo 7X h	A.1.G.GUADFGI	ARF	Ameri	-1	GUADFGI	1.7E+08	Sales Order Pick	6/29/2015 6:53	GUADSTAG	ARF	GUADSTA	Each
15	94000-20	Case IH FM-750 Di	GUADSTAG.1.1.1	ARF	Ameri	57	GUADSTAG	1.7E+08	Internal Order Pick	7/24/2015 8:36	GUADFGI	ARF	B.1.B.GUA	Each
16	88056-00	GeoExplorer 6000	GUADSTAG.1.1.1	ARF	Ameri	-3	GUADSTAG	1.7E+08	Subinventory Transfer	6/4/2015 11:28	GUADSTAG	ARF	GUADPCK	Each
17	65535	Cable, Add-on, 3-	A.1.A.GUADFGI	ARF	Ameri	-6	GUADFGI	1.7E+08	Internal Order Pick	4/2/2015 8:26	GUADSTAG	ARF	GUADSTA	Each
18	88170-11	PACKOUT, Keanu	A.1.A.GUADPACK	ARF	Ameri	21	GUADPACK	1.6E+08	PO Receipt	11/6/2014 9:45			...	Each
19	81618	PAD, DOUBLE SID	A.1.A.GUADPACK	ARF	Ameri	-100	GUADPACK	1.7E+08	WIP component issue	6/4/2015 15:48			...	Each
20	99100-21	Trimble Pro 6T Re	GUADPCK.1.1.1	ARF	Ameri	-2	GUADSTAG	1.8E+08	Sales order issue	9/17/2015 13:14			...	Each
21	70956-00	3.5" Clear Touch S	GUADSTAG.1.1.1	ARF	Ameri	2	GUADSTAG	1.6E+08	Sales Order Pick	12/5/2014 15:03	GUADFGI	ARF	A.1.A.GUA	Each
22	92000-60	EZ Guide 250, Nev	GUADPCK.1.1.1	ARF	Ameri	1	GUADSTAG	1.7E+08	Subinventory Transfer	5/27/2015 6:51	GUADSTAG	ARF	GUADSTA	Each
23	70956-00	3.5" Clear Touch S	GUADPCK.1.1.1	ARF	Ameri	-75	GUADSTAG	1.7E+08	Int Order Intr Ship	3/31/2015 12:15	B-FGI	TBV	...	Each
24	56237-91	Antenna, AG, 5vd	GUADSTAG.1.1.1	ARF	Ameri	8	GUADSTAG	1.7E+08	Internal Order Pick	3/17/2015 11:00	GUADFGI	ARF	A.1.A.GUA	Each

Figure 6.1: Sample of the inventory transaction data set

model is composed of two parts: autoregressive model (AR) and moving average model (MA).

Define AR(p) the autoregressive model of order p. AR(p) can be written as:

$$X_t = c + \sum_{i=1}^{p} \varphi_i X_{t-i} + \varepsilon_t \tag{6.1}$$

where $\varphi_1, \ldots, \varphi_p$ are parameters, c is a constant, and the random variable ε_t is white noise.

Define MA(q) the moving average model of order q. MA(q) can be written as:

$$X_t = \mu + \varepsilon_t + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} \tag{6.2}$$

where θ_i are the parameters of the model, μ is the expectation of X_t.

Define ARMA(p,q) the autoregressive moving average with p AR terms and q MA terms. ARMA(p,q) can be written as:

$$X_t = c + \varepsilon_t + \sum_{i=1}^{p} \varphi_i X_{t-i} + \sum_{i=1}^{q} \theta_i \varepsilon_{t-i} \tag{6.3}$$

6.3.1.2 Data Preprocess

Originally in the system, one inventory transaction is recorded with the moment that it occurs. Therefore before proceeding to predicting inventory shipments it is necessary to determine the time scale of data points. In [71] [72], data points are measured as “per hour”. In our ARMA implementation, we define the time scale as “per day”, that is proven to optimize prediction accuracy according to our observations.

To obtain the shipment quantity for each day demands accumulating the quantity of each shipment transaction occurred in the same day, as in equation 4:

$$N_m^j = \sum_{i=0}^{n_j} N_m^{ji} \tag{6.4}$$

where N_m^j denotes the daily shipment quantity of product m in day j, n_j denotes the number of shipments occurred in day j for product m, and N_m^{ji} denotes each record of shipment quantity in day j for product m.

Figure 6.2 demonstrates the extracted shipment quantity records for product 92000-20 in STAG by days. They do not indicate a general periodic pattern, because the relevant business activities can be intricate and intermittent. For instance, One factor that affects the shipments is sales orders, that are not consistent in nature: some sales orders tend to be pooled when it approaches the ends of quarters; some sales orders occasionally have varied order quantities.

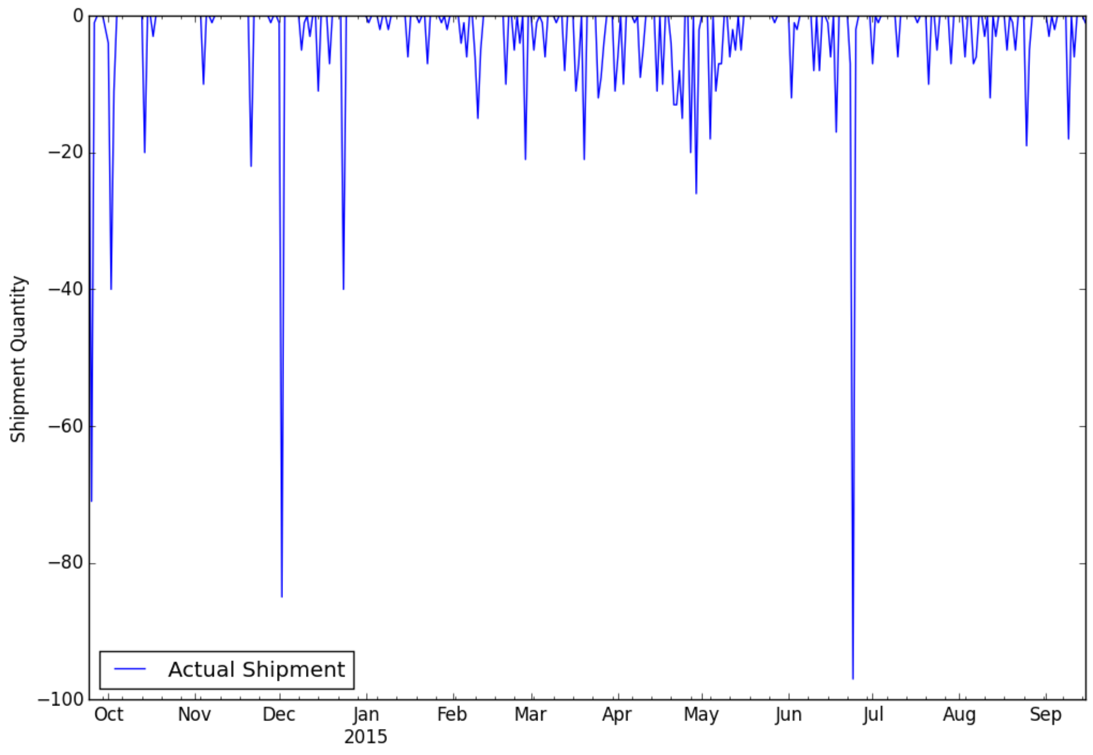


Figure 6.2: Annual shipments of 92000-20

6.3.1.3 ARMA Evaluation

The data set that is used for our evaluation starts from September 25, 2014 and ends on September 24, 2015. In our implementation, we select the inventory shipments that occurred before June 5, 2015 as training set, and later ones as the testing set. For simplicity, in this chapter, we do not apply a validation set to determine the parameter settings that achieve the optimal performance. Instead, we estimate some groups of parameters that can approach the optimal prediction accuracy and evaluate the prediction accuracy under each of them. Table 6.2 demonstrates the evaluation results for 10 products with high shipment frequencies by the ARMA model under different settings of the order p (q is not defined in this case because changing q merely impact on the performance according to our observation). Root-mean-square-error (RMSE) is the metric to depict how different the predicted values are from the actual values, that is defined as

$$\mathrm{RMSE}(\hat{\theta}) = \sqrt{\mathrm{MSE}(\hat{\theta})} = \sqrt{\mathrm{E}((\hat{\theta} - \theta)^2)}. \tag{6.5}$$

It can be observed from Table 6.2 that generally, changing p does not change the prediction accuracy by very much. Moreover, the data set with higher deviations among themselves tend to have worse prediction accuracy.

Products	p=3	p=10	p=20	Effective Days	Data Deviation
92000-20	9.94	9.94	9.93	110	84.25
94000-60	12.69	12.74	12.89	101	139.54
88014-00	7.46	7.41	7,42	87	37.07
88180-01	3.36	3.36	3.41	70	42.36
92000-60	18.31	18.35	18.49	98	152.27
94000-00	11.65	11.65	11.66	136	304.68
92000-00	3.10	3.10	3.10	93	25.27
88180-02	4.10	4.11	4.11	57	16.57
88004-04	8.07	8.02	8.08	123	85
88180-04	5.31	5.31	5.30	70	49.95

Table 6.2: RMSE under different p and q settings for ARMA implementations

6.3.2 The Methodology with KNN Algorithm

It can be observed from Table 6.2 that the shipment days of products range from 1/7 to 1/2 year. This type of the data sets can be considered sparse. To handle the prediction for sparse data, a new K Nearest Neighbor (KNN) algorithm is proposed and proven to be efficient [71] [72]. In this chapter, their insights of connecting past data points and future data points are adopted while the data points configuration are modified for a better prediction accuracy and response time performance.

6.3.2.1 Data Preprocess

It is assumed that there is a function relation between the future inventory transactions and the past inventory transactions. Assume for any product, $S(t)$ denotes the actual inventory shipment at time t, $\hat{S}(t)$ denotes the prediction of inventory shipment at time t, and $\epsilon(t)$ denotes the noise that occurs at time t. The relation can be represented as

$$\hat{S}(t) = F(S(t-i), \epsilon(t)), i \in \{1, 2, ...\} \tag{6.6}$$

Particularly in this chapter, by defining the time scale to be "per hour", we assume that the actual inventory transaction at time t is relevant to the data points that are integer multiples of 24 hours earlier, e.g., $t - 24$ hours, $t - 48$ hours, etc. It is reasonable in practice that inventories, especially the ones with high-quality management tend to deal with inventory transactions by following certain patterns. Thereby it is possible that shipments for products occur around certain time during a day. These patterns provide the feasibility to associate the future inventory transactions with the past ones that occur exactly integer multiple of 24 hours earlier.

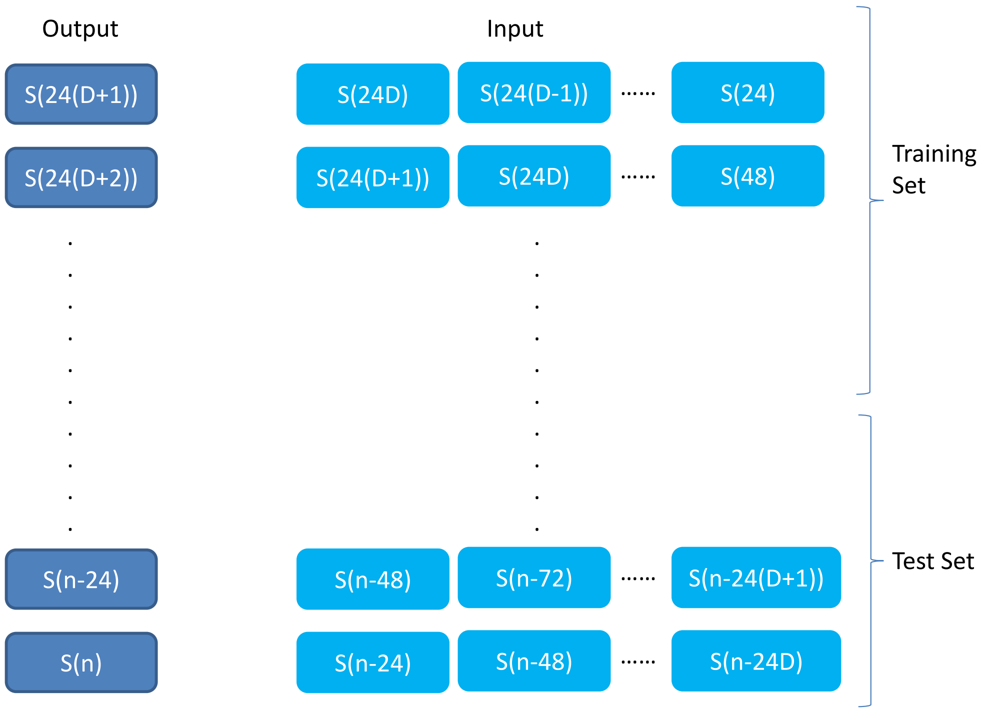


Figure 6.3: Group past inventory shipments that differ by 24 hours as input data and pair them with the future inventory shipments as output data

Figure 6.3 demonstrates how past inventory shipments are connected with future inventory shipments. Typically, future inventory shipments are defined as output data set. For each future inventory shipment, the ones that occur by every 24 hours earlier are defined as its input data set. The depth of backtracing the input data set, i.e., the size of each input data set is defined as D. The entire data set is divided into training set with earlier inventory shipments and test set with later ones.

Our definition for input and output data sets differs from [71] in that the data points in [71] is defined as the vector of the events that occur in the past 24 hours while ours is simply a single value due to our assumption for inventory shipment pattern mentioned above. Our solution advances in reducing the response time performance.

6.3.2.2 Primitive KNN Algorithm

In our implementations, define $y(t)$ as the output data, i.e., the predicted inventory shipment $S(t)$ that occurs in time t, and $x(t)$ as the input data $\{S(t-24)$, $S(t-48)$,...,$S(t-24D)\}$. According to Figure 6.3, if there are N days in our data set, there is $(N-D+1)$ input-output pairs and $24N$ total data points.

Originally, the KNN algorithm was used for classification with given class labels. In this chapter, KNN is modified such that, for a given future inventory shipment, its input is computed with all the inputs in training data for dissimilarities and the K smallest ones are selected as its nearest neighbors for averaging their outputs to be the predicted inventory shipment. Particularly in Algorithm 3, to find an estimate of one future inventory shipment, namely *$y(t_{s*})$*, where t_{s*} is an instance of test set t_s, the dissimilarity between

Algorithm 3 Primitive KNN Algorithm

Input:

$\boldsymbol{x}(t_r)$, $y(t_r)$, $x(t_{s*})$, k

Output:

$y(t_{s*})$

```
1: for j $\in t_r$ do
2:     dis[j] = $\|x(t_{s*}) - x(j)\|$;
3: end for
4: for i $\in \{1,...,k\}$ do
5:     idx[i] = index of i$^{th}$ smallest dis
6: end for
7: $y(t_{s*}) = (1/k)\sum_{i\in\{1,...,k\}} y(idx[i])$
```

its input $x(t_{s*})$ and all the data points of the training set $\boldsymbol{x}(t_r)$ needs to be computed. Here Euclidean Distance is applied to measure the dissimilarity. Then the K nearest $x(t_r)$ to $x(t_{s*})$ can be determined and at last, $y(t_{s*})$ is derived by averaging all the corresponding $y(t_r)$.

6.3.2.3 Primitive KNN Evaluation

In our Primitive KNN implementation, the prediction accuracy is found to be optimal when K equals 1. The evaluations are implemented in terms of various Ds $(20, 30, 40, 50)$. Similar with the ARMA evaluation, we do not apply validation sets for simplicity.

It can be observed from Table 6.3 and Table 6.2 that the prediction by Primitive KNN algorithm can always provide better accuracy compared with the prediction by ARMA algorithm by selecting proper Ds.

Products	D=20	D=30	D=40	D=50
92000-20	4.53	16.51	1.65	0.84
94000-20	14.14	1.02	6.03	5.36
94000-60	5.84	2.69	2.08	3.33
88014-00	6.99	7.10	6.67	6.74
88180-01	3.45	2.31	2.06	2.01
92000-60	12.72	5.86	5.44	5.41
94000-00	5.62	8.04	5.20	3.53
92000-00	3.46	3.36	2.88	2.09
88180-02	4.00	4.01	3.07	3.13
88004-04	5.83	8.96	4.31	3.91
88180-04	8.02	7.22	7.13	3.60

Table 6.3: RMSE under Primitive KNN implementations

6.3.2.4 Time-Efficient KNN

In KNN implementations, to find K nearest neighbors, a routing solution is to simply sort all the dissimilarities in a certain manner and select the K smallest ones. If the size of the dissimilarities is n, the optimal time complexity to find out these K smallest dissimilarities is $O(n\lg n)$. In our approach, note that optimal accuracy can be achieved when K equals 1, that is, for any future inventory shipment, namely S_f, its prediction is computed as the inventory shipment whose input is nearest to S_f's. To quickly find this smallest dissimilarity to S_f's input, an improvement based on the Primitive KNN algorithm is proposed in this section and the time complexity to find this smallest dissimilarity can be significantly reduced.

In Algorithm 4, two new fields are included as inputs, where Min stores the updated minimum dissimilarity and the $Minindex$ stores the index

Algorithm 4 Time-Efficient KNN Algorithm

Input:

$\boldsymbol{x}(t_r)$, $x(t_{s*})$, Min = MAX_VALUE, $Minindex$ = -1

Output:

$y(t_{s*})$

```
1: for j ∈ $t_r$ do
2:     dis[j] = $\|x(t_{s*}) - x(j)\|$;
3:     if dis[j] < $Min$ then
4:         $Min$ = dis[j]
5:         $Minindex$ = j
6:     end if
7: end for
8: $y(t_{s*}) = y(Minindex)$
```

of the inventory shipment that renders the minimum dissimilarity. Given the input of a future inventory shipment $x(t_{s*})$, each time when its dissimilarity (e.g., Euclidean Distance) to the inputs of the training set is computed, the dissimilarity is compared with Min. If the dissimilarity is smaller, then Min is updated as that dissimilarity with $Minindex$ updated. This process iterates till all the computations are completed and Min at that time is guaranteed to be the smallest dissimilarity between $x(t_{s*})$ and $\boldsymbol{x}(t_r)$ with $Minindex$ representing the index of the inventory shipment with that dissimilarity.

Algorithm 4 guarantees finding the minimum dissimilarity along with the computations of dissimilarities, where no specific step for sorting is needed.

6.3.3 The Methodology with Approximate Nearest Neighbor Algorithm

Although the Time-Efficient KNN algorithm improves the response time performance compared with the Primitive KNN algorithm, it still can be inefficient in real-world applications, with the reasons listed as follows.

- In real-world applications, a validation set, that validates the optimal configurations of parameters (such as the number of nearest neighbors K, the time depth D, etc., in our approach), needs to be implemented before running test sets. Generally in a validation set, multiple preset groups of parameter settings are applied to the training set, e.g., $(K = 1, D = 1)$, $(K = 1, D = 2)$, $(K = 2, D = 1)$. Then the ones with the optimal prediction accuracy are selected as the parameters for test sets. This process can be significantly time-consuming.

- While nowadays data engineers tend to upload data sets to clouds, e.g., Amazon EC2, for strong computation power to improve the response time performance, business owners might be reluctant to do so because of the security issues. More likely, computations are conducted in physical servers owned by companies, that can vary widely in computation power. Therefore, it can be risky to apply the cluster of KNN algorithms to build Business Intelligence applications that are sensitive with response times.

To handle these challenges, in this section I propose a new algorithm that approximates the nearest neighbor algorithm for prediction but that

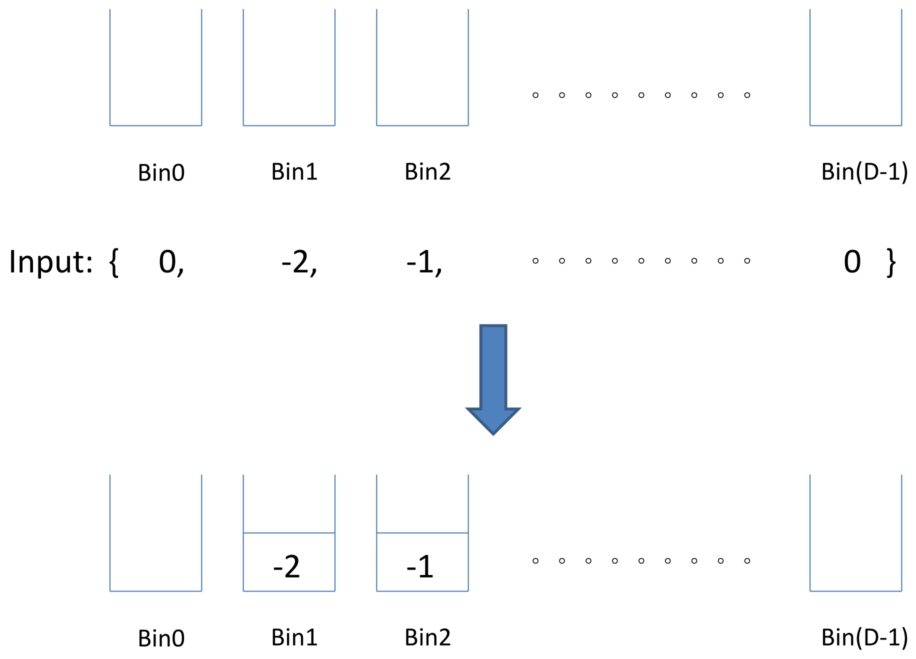


Figure 6.4: Placing non-zero elements of an input inventory shipment to bins with the same dimensions

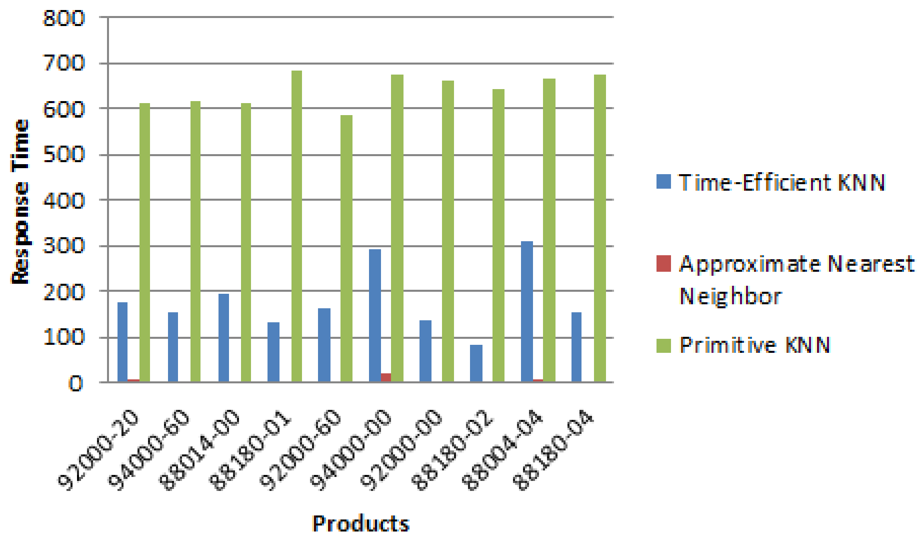
demands much less response time so that it can be completely under the company's control. Note that in many cases inventory shipments occur in daytime and the effective days of inventory shipments for any product do not exceed half a year. It can be concluded that there are large amounts of inventory shipment inputs are all zeros, that causes redundant computations. Moreover, it can be observed that for most of non-all-zero inputs, they have only a few non-zero elements, that indicates there are amounts of redundant dissimilarity

Algorithm 5 Approximate Nearest Neighbor Algorithm

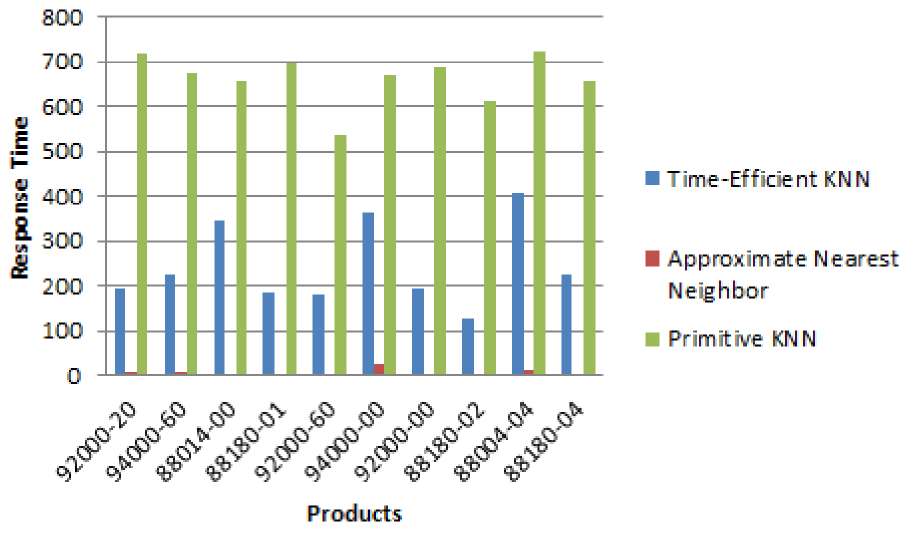
```
Input:
$\boldsymbol{x}(t_r)$, $y(t_r)$, $x(t_{s*})$, $Min$ = Integer.MAX_VALUE, $Minindex$ = -1
Output:
$y(t_{s*})$

1: Build bins and place non-zero elements to them from $\boldsymbol{x}(t_r)$;
2: identify non-zero elements in $x(t_{s*})$;
3: identify the bins with the corresponding indices, all the indices included
   within, namely $b_i$, and the corresponding input inventory transactions,
   namely $x(b_i)$;
4: for j $\in b_i$ do
5:     dis[j] = $\|x(t_{s*})$ - $x(j)\|$;
6:     if dis[j] $< Min$ then
7:         if dis[j] $< \|x(t_{s*})\|$ then
8:             $Min$ = dis[j]
9:             $Minindex$ = j
10:        else
11:            $Min = \|x(t_{s*})\|$
12:            $Minindex$ = idx({0,...,0})
13:        end if
14:    end if
15: end for
16: $y(t_{s*}) = y(Minindex)$
```

computations that occur between two zeros. These redundancies imply that, if they can be detected and discarded, it is possible that response time can be significantly reduced.

This algorithm, aiming at easily detecting redundant computation between inputs, is initiated by creating bins with the same dimensions as the depth D. In the training set, all the input inventory shipments place their non-zero elements into the bins with the corresponding indices. This process

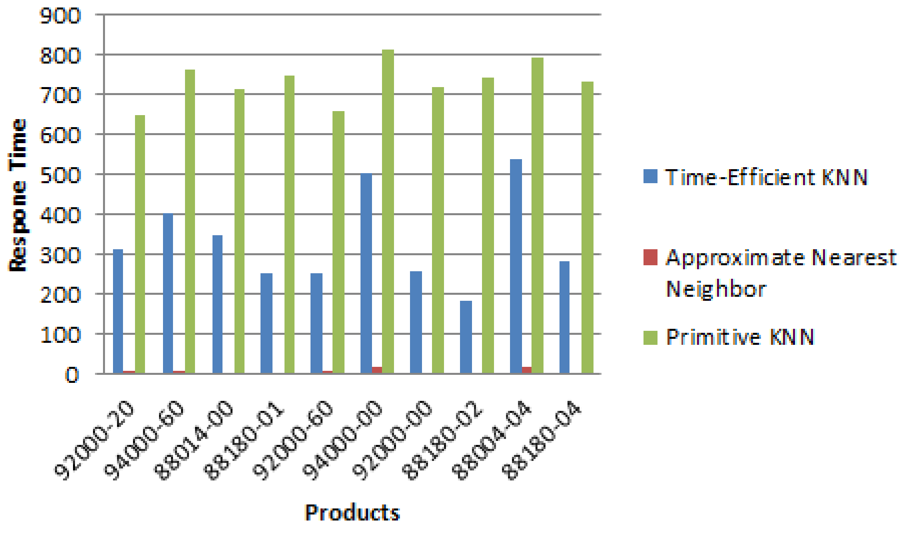


(a) *D =20*

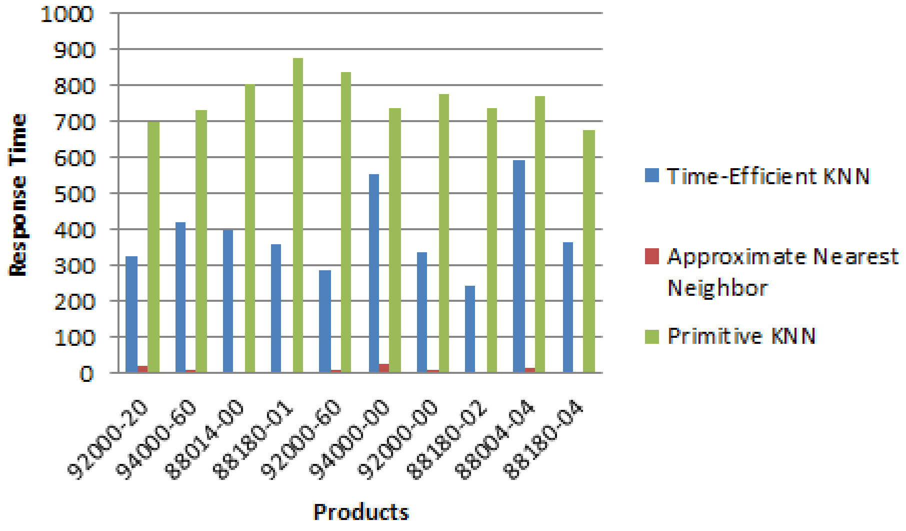


(b) *D = 30*

Figure 6.5: The evaluation results showing the computation time of the Approximate Nearest Neighbor, Primitive and Time-Efficient KNN implementations when D = 20 and 30



(a) *D =40*



(b) *D = 50*

Figure 6.6: The evaluation results showing the computation time of the Approximate Nearest Neighbor, Primitive and Time-Efficient KNN implementations when D = 40 and 50

can be illustrated in Figure 6.4, where D bins are built. Assume an input inventory shipment $\{0, -2, -1, ..., 0\}$ and its non-zero elements -2 and -1 are placed in the bins with the same indices as theirs.

Given a future inventory shipment S_f, to compute dissimilarities between it and all the input inventory shipments in training set can be simplified in our algorithm. First, the non-zero elements of S_f are selected. For each of these non-zero elements, their corresponding bins are selected and the indices of all the elements in these bins are identified. Then the computation of the dissimilarity between S_f and the input inventory shipments is simplified as to compute the dissimilarity between S_f and the input inventory shipments with the identified indices, as in Algorithm 5. Note that a baseline input-output pair, that is an all-zero input and 0 output, is included for computation to determine the minimum dissimilarity.

The evaluation results of response time performance between the Approximate Nearest Neighbor, Time-Efficient KNN, and Primitive KNN algorithms is listed in Figure 6.5 and 6.6. Overall, the Time-Efficient KNN algorithm advances in response time compared with the Primitive KNN algorithm. Specifically, when D is smaller, the performance gain is bigger. For instance, when D is 20, the Time-Efficient KNN performs from 53.9% to 87.1% better than the Primitive KNN algorithm. On the other hand, the Approximate Nearest Neighbor algorithm achieves a significant advantage in response time performance compared with the Time-Efficient KNN algorithm. For instance, when D is 20, the Approximate Nearest Neighbor algorithm saves from 92.9%

to 98.9% response time over Time-Efficient KNN algorithm. It can be concluded that the Approximate Nearest Neighbor algorithm is much more time-efficient than the others. In real-world applications, this advantage can be expected to be more significant when validation sets are included.

Products	D=20	D=30	D=40	D=50
92000-20	11.03	19.37	10.33	10.33
94000-60	13.69	12.72	12.62	12.92
88014-00	8.14	8.24	7.88	7.94
88180-01	3.55	3.55	3.55	3.53
92000-60	22.02	18.89	18.84	18.83
94000-00	10.37	12.86	11.87	11.84
92000-00	3.93	3.90	3.50	3.07
88180-02	4.09	4.09	4.34	4.39
88004-04	10.88	10.58	10.59	10.61
88180-04	8.55	8.14	8.05	5.26

Table 6.4: RMSE under Approximate Nearest Neighbor implementations

Methods	ARMA	KNN	Approximate NN
Average RMSE	8.39	3.32	8.65

Table 6.5: Average RMSE under different methods

The results of prediction accuracy by the Approximate Nearest Neighbors is demonstrated in Table 6.4. To better understand the performances of all the methodologies, I average the optimal prediction accuracy for each product as in Table 6.5. It can be shown that KNN algorithms have the best prediction accuracy, and Approximate Nearest Neighbor and ARMA perform similarly. As a trade-off, the Approximate Nearest Neighbor algorithm does not perform as well as KNN algorithms in prediction accuracy, while it is much more time-efficient.

To further illustrate the response time performances among all the approaches, another set of evaluations are conducted where the validation sets are included. 70% of the earlier data are used as the training set, 10% of the later data are used as the validation set to determine the optimal parameter settings for testing data. Specifically, starting from 20 and incrementing by 5 for the dimension D, 10 groups of pre-set parameter settings are conducted to be selected for the optimal D that is used for testing data.

Products	Time-Efficient KNN	Approximate Nearest Neighbor
92000-20	3497.2	20.97
94000-60	4080.6	11.89
88014-00	3510.3	5.95
88180-01	2582.3	5.36
92000-60	2704.6	7.37
94000-00	5367.3	25.77
92000-00	2075.2	7.71
88180-02	1674.9	2.75
88004-04	5682.5	17.78
88180-04	2617.3	4.74

Table 6.6: Response time (seconds) for Time-Efficient KNN with validation sets and Approximate Nearest Neighbor implementations

Table 6.6 records the response time of the Time-Efficient KNN and the Approximate Nearest Neighbor, where 10 groups of dimension Ds are considered for validation sets for the Time-Efficient KNN. Table 6.7 records the optimal RMSE and the corresponding D of the Time-Efficient KNN by involv-

Products	RMSE	Optimal D
92000-20	0.84	50
94000-60	2.08	40
88014-00	6.67	40
88180-01	1.86	65
92000-60	5.26	60
94000-00	3.53	50
92000-00	2.09	50
88180-02	3.07	40
88004-04	2.28	55
88180-04	3.60	50

Table 6.7: RMSE and the corresponding D of the Time-Efficient KNN with validation sets

ing validation sets. Table 6.6 indicates that the response time of Time-Efficient KNN algorithm is around 200 to 700 times more than that of Approximate Nearest Neighbor algorithm. It can be speculated that with more amount of data points included and more groups of parameter settings for optimal parameter selections, the response time of Time-Efficient algorithm could be even larger than that of Approximate Nearest Neighbor algorithm. Moreover, it seems that in this thesis, the fact that the optimal Ds for these products range from 40 to 65 implies for these applications, the computation loads are possibly reduced, especially when the computations on these data sets are repeated with these optimal Ds being apriori knowledge. However, in real-world applications, even when these data are repeatedly used for computations, it is much likely that they are included as a subset together with other data and the optimal Ds could be different. Therefore, it is inevitable to conduct the validation sets in each run of the computations and the computation loads are not expected to be lowered even if some of the data sources are repeated used.

All of the evaluations are conducted with Python 2.0 on an Intel Core i-7 CPU at 3.40 GHz with 16 GB RAM.

Chapter 7

Conclusion

My dissertation develops modeling techniques to provide solutions for business process optimization and evaluation, and its data analysis. Specifically, I develop a data-centric business process modeling technique and approaches to optimize the temporal performance of the data-centric business processes by reconstructing them. To evaluate these approaches, I build a symbolic process generator to stochastically generate symbolic data-centric business processes that can be used to analyze their properties, i.e., to evaluate my temporal optimization techniques in this thesis. Finally, I present a framework to forecast typical business process data, i.e., inventory shipments and have it evaluated with real-world data sets extracted from Oracle EBS systems.

My data-centric business process modeling technique explicitly models composite business activities that is specified as a tuple of components: data, human actor, and atomic activity. Data is modeled to reflect its states and life cycle in workflows. Moreover, workflows of business processes are represented by defining the concepts of subprocesses, clusters, and policies of their placements.

The time-optimization approach reconstructs business process by modifying the execution order of business activities, that is realized by relocating business activities from their original subprocesses to others. This approach is initiated by determining whether a business activity is eligible for relocation in a business process. One business activity is defined to be eligible for relocation to a subprocess only when it follows the patterns of being forwardly moveable, backwardly moveable, or loosely-parallelled moveable to the subprocess. Accordingly, the approaches that optimizes temporal performances of business process under different circumstances are developed.

To evaluate these approaches, G-DCBP is developed to generate symbolic business processes. It generates process components (i.e., data, activities, subprocesses, and clusters) and creates a hierarchy that incorporates them into data-centric business processes. Its inputs are simply the value (stochastic and deterministic) of the process components. The evaluation results reflect the degree of temporal improvements of business processes in different scenarios and reveal that the size of business processes and the data dependency rate have the significant impacts on the temporal improvements.

At last, I pay attentions to granular data of business processes, that is inventory shipments. In this thesis, I present a framework to predict shipments using historical data from Oracle EBS system, which is widely used in enterprises for operation management. A set of Nearest-Neighbor-based algorithms are developed for different performance metrics, including prediction accuracy and response time, that are both important performance metrics in the domain of business intelligence.

Appendices

Appendix A

An Example of Data-Centric Business Process

```
activity  apartment_purchasing_application
   {
   role  applicant, government_employee, government_officer;
   initial_data  apartment_application_form, contract;
   input_data  empty
   global_data  government_ID, bank_statement, credit_history;
   consumed_data  apartment_application_form, contract;
   atomic_activity  <fill_out_apartment_application_form>,
                    <submit_application_form>,
                    <verify_identification_and_qualification>;
                    <approve_apartment_application_form>;
   output_data  approved_apartment_application_form,
   final_data  empty;
   }

activity  loan_application
   {
   role  applicant, bank_employee, bank_manager;
   initial_data  loan_application_form;
   input_data  empty;
   global_data  government_ID, bank_statement, credit_history;
   consumed_data  loan_application_form;
   atomic_activity   <fill_out_loan_application_form>,
                     <submit_loan_application_form>,
                     <approve_loan_application_form>;
   output_data  approved_loan_application_form,
   final_data  empty;
   }

activity down_payment
   {
```

```
   role applicant, bank_employee;
   input_data  approved_apartment_application_form,
   global_data  government_ID,
                bank_statement,
                credit_history;
   consumed_data  empty;
   atomic_activity  <submit_approved_apartment_application_form>,
                    <pay_down_payment>;
   output_data  approved_apartment_application_form;
   final_data  empty;
   }

activity  apartment_evaluation
   {
   role  applicant, bank_employee, government_employee;
   initial_data  empty;
   input_data   approved_loan_application_form,
   global_data  archived_apartment_info;
   consumed_data  empty;
   atomic_activity  <check_archived_apartment_info>,
                    <evaluated_apartment_value>;
   output_data  approved_loan_application_form,
                apartment_value;
   final_data  empty;
   }

activity  asset_evaluation
   {
   role  applicant, bank_employee, government_employee;
   initial_data  empty;
   input_data   approved_loan_application_form,
                apartment_value;
   global_data  government_ID, bank statement, credit history;
   consumed_data  apartment_value;
   atomic_activity  <submit_bank_statement_and_credit_history>,
                    <verify_identification_and_qualification>;
                    <verify_bank_statement_and_credit_history>,
                    <evaluate_qualification_of_asset>,
                    <make_decisions_after_evaluation>;
   output_data  approved_loan_application_form,
                estimated_loan_rate,
   final_data  empty;
   }

activity  insurance
   {
   role  applicant, insurance_employee, insurance_manager;
   initial_data  empty;
   input_data  approved_loan_application_form,
               approved_apartment_application_form,
   global_data  government_ID,
                bank_statement,
                credit_history,
                insurance_rules;
   consumed_data  empty;
   atomic_activity  <submit_materials>,
                    <evaluate_insurance_policy>,
                    <offer_insurance_policy>,
                    <negotiate_insurance_policy>,
                    <finalize_insurance_policy>;
   output_data   insurance_policy,
                 approved_loan_application_form,
                 approved_apartment_application_form;
   final_data  empty;
   }

activity  issue_certificate
   {
   role  applicant, government_employee, government_officer;
   initial_data  empty;
   input_data  approved_loan_application_form,
               approved_apartment_application_form,
               insurance_policy;
   global_data  government_ID;
   consumed_data  empty;
   atomic_activity  <submit_materials>,
                    <verify_materials>,
                    <approve_certificate>,
                    <issue_certificate>;
   output_data  empty;
```

```
   final_data  approved_loan_application_form,
               approved_apartment_application_form,
               insurance_policy;
   }

activity  public_notification
   {
   role  government_employee, government_officer;
   initial_data  empty;
   input_data  approved_loan_application_form,
               approved_apartment_application_form,
               insurance_policy;
   global_data  empty;
   consumed_data  empty;
   atomic_activity  <verify_materials>,
                    <post_public_notification>;
   output_data  empty;
   final_data  approved_loan_application_form,
               approved_apartment_application_form,
               insurance_policy;
   }

activity  document_archive
   {
   role  government_employee, government_officer;
   initial_data  empty;
   input_data  approved_loan_application_form,
               approved_apartment_application_form,
               insurance_policy;
   global_data  empty;
   consumed_data  empty;
   atomic_activity  <verify_materials>,
                    <archive_materials>;
   output_data  empty;
   final_data  approved_loan_application_form,
               approved_apartment_application_form,
               insurance_policy;
   }
```

Bibliography

[1] Yutian Sun and Jianwen Su. Computing degree of parallelism for bpmn processes. In *Proceedings of the 9th international conference on Service-Oriented Computing*, ICSOC'11, pages 1–15, Berlin, Heidelberg, 2011. Springer-Verlag.

[2] Object Management Group (OMG). Business process model and notation (bpmn) version 2.0. Technical report, jan 2011.

[3] T.J. Schriber. *Fundamentals of flowcharting.* Wiley, 1969.

[4] Jerry L. Jones. *Structured programming logic - a flowcharting approach.* Prentice Hall, 1985.

[5] Finn Jorgensen. Overview of function modelling - idef0. In Heimo H. Adelsberger, Jiri Lazansky, and Vladimir Marik, editors, *Information Management in Computer Integrated Manufacturing*, volume 973 of *Lecture Notes in Computer Science*, pages 340–354. Springer, 1995.

[6] Mounira Harzallah, Giuseppe Berio, and Andreas L. Opdahl. Incorporating idef3 into the unified enterprise modelling language. In *EDOCW*, pages 133–140. IEEE Computer Society, 2007.

[7] Wolfgang Reisig. *A primer in Petri net design.* Springer Compass International. Springer, 1992.

[8] Matjaz B. Juric. *Business Process Execution Language for Web Services BPEL and BPEL4WS 2nd Edition.* Packt Publishing, 2006.

[9] Dave Ings, Luc Clment, Dieter Konig, Vinkesh Mehta, Ralf Mueller, Ravi Rangaswamy, Michael Rowley, and Ivana Trickovic. Ws-bpel extension for people (bpel4people) specification version 1.1. OASIS Committee Specification, August 2010.

[10] M. Bischof, O. Kopp, T. van Lessen, and F. Leymann. Bpelscript: A simplified script syntax for ws-bpel 2.0. In *Software Engineering and Advanced Applications, 2009. SEAA '09. 35th Euromicro Conference on*, pages 39–46, 2009.

[11] Alin Deutsch, Richard Hull, Fabio Patrizi, and Victor Vianu. Automatic verification of data-centric business processes. In *Proceedings of the 12th International Conference on Database Theory*, ICDT '09, pages 252–267, New York, NY, USA, 2009. ACM.

[12] Wil M. P. van der Aalst. Verification of workflow nets. In *Proceedings of the 18th International Conference on Application and Theory of Petri Nets*, ICATPN '97, pages 407–426, London, UK, UK, 1997. Springer-Verlag.

[13] Wil M. P. van der Aalst. Workflow verification: Finding control-flow errors using petri-net-based techniques. In *Business Process Management, Models, Techniques, and Empirical Studies*, pages 161–183, London, UK, UK, 2000. Springer-Verlag.

[14] Gregor Engels and Martin Assmann. Service-oriented enterprise architectures: Evolution of concepts and methods. In *EDOC*. IEEE Computer Society, 2008.

[15] C. Gerth, M. Luckey, J.M. Kuster, and G. Engels. Detection of semantically equivalent fragments for business process model change management. In *Services Computing (SCC), 2010 IEEE International Conference on*, pages 57–64, 2010.

[16] C. Gerth, M. Luckey, J.M. Kuster, and G. Engels. Precise mappings between business process models in versioning scenarios. In *Services Computing (SCC), 2011 IEEE International Conference on*, pages 218–225, 2011.

[17] Carla Marques Pereira and Pedro Sousa. A method to define an enterprise architecture using the zachman framework. In *Proceedings of the 2004 ACM symposium on Applied computing*, SAC '04, pages 1366–1371, New York, NY, USA, 2004. ACM.

[18] Stefan Schulte, Kay Kadner, Nicolas Repp, and Ralf Steinmetz. Applied service engineering for single services and corresponding service landscapes. In Robert C. Nickerson and Ramesh Sharda, editors, *AMCIS*, page 472. Association for Information Systems, 2009.

[19] Gregor Engels, Andreas Hess, Bernhard Humm, Oliver Juwig, Marc Lohmann, Jan-Peter Richter, Markus VoB, and Johannes Willkomm.

A method for engineering a true service-oriented architecture. In Jose Cordeiro and Joaquim Filipe, editors, *ICEIS (3-2)*, pages 272–281, 2008.

[20] Michael J. Berry and Gordon Linoff. *Data Mining Techniques: For Marketing, Sales, and Customer Support.* John Wiley & Sons, Inc., New York, NY, USA, 1997.

[21] E.W.T. Ngai, Li Xiu, and D.C.K. Chau. Application of data mining techniques in customer relationship management: A literature review and classification. *Expert Systems with Applications*, 36(2, Part 2):2592 – 2602, 2009.

[22] Matthew Richardson and Pedro Domingos. Mining knowledge-sharing sites for viral marketing. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, pages 61–70, New York, NY, USA, 2002. ACM.

[23] Alex G. Büchner and Maurice D. Mulvenna. Discovering internet marketing intelligence through online analytical web usage mining. *SIGMOD Rec.*, 27(4):54–61, December 1998.

[24] Sven F. Crone, Stefan Lessmann, and Robert Stahlbock. The impact of preprocessing on data mining: An evaluation of classifier sensitivity in direct marketing. *European Journal of Operational Research*, 173(3):781 – 800, 2006.

[25] Siddhartha Bhattacharyya. Evolutionary algorithms in data mining: Multi-objective performance modeling for direct marketing. In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '00, pages 465–473, New York, NY, USA, 2000. ACM.

[26] Shu-Hsien Liao, Jen-Lung Chen, and Tze-Yuan Hsu. Ontology-based data mining approach implemented for sport marketing. *Expert Systems with Applications*, 36(8):11045 – 11056, 2009.

[27] Lean inventory management. `http://www.waspbarcode.com/buzz/lean-inventory-management/`.

[28] Karl Inderfurth. Safety stock optimization in multi-stage inventory systems. *International Journal of Production Economics*, 24(1?):103 – 113, 1991.

[29] Alessandro Persona, Daria Battini, Riccardo Manzini, and Arrigo Pareschi. Optimal safety stock levels of subassemblies and manufacturing components. *International Journal of Production Economics*, 110(1?):147 – 159, 2007.

[30] Stephen C. Graves and Sean P. Willems. Optimizing strategic safety stock placement in supply chains. *Manufacturing & Service Operations Management*, 2(1):68–83, January 2000.

[31] Yi-Chung Hu, Ruey-Shun Chen, and Gwo-Hshiung Tzeng. Finding fuzzy classification rules using data mining techniques. *Pattern Recognition Letters*, 24(1?):509 – 519, 2003.

[32] Karthik Sourirajan, Leyla Ozsen, and Reha Uzsoy. A genetic algorithm for a single product network design model with lead time and safety stock considerations. *European Journal of Operational Research*, 197(2):599 – 608, 2009.

[33] Srinivas Talluri, Kemal Cetin, and A.J. Gardner. Integrating demand and supply variability into safety stock evaluations. *International Journal of Physical Distribution & Logistics Management*, 34(1):62–69, 2004.

[34] Enwang Zhou and Alireza Khotanzad. Fuzzy classifier design using genetic algorithms. *Pattern Recognition*, 40(12):3401 – 3414, 2007.

[35] Ramakrishnan Ramanathan. {ABC} inventory classification with multiple-criteria using weighted linear optimization. *Computers & Operations Research*, 33(3):695 – 700, 2006.

[36] Fariborz Y Partovi and Murugan Anandarajan. Classifying inventory using an artificial neural network approach. *Computers & Industrial Engineering*, 41(4):389 – 404, 2002.

[37] Hafedh Mili, Guy Tremblay, Guitta Bou Jaoude, Éric Lefebvre, Lamia Elabed, and Ghizlane El Boussaidi. Business process modeling lan-

guages: Sorting through the alphabet soup. *ACM Comput. Surv.*, 43(1):4:1–4:56, December 2010.

[38] R.S.Ruth Sara Aguilar-Saven. Business process modelling: Review and framework. *International Journal of Production Economics*, 90(2):129–149, july 2004.

[39] George M. Giaglis. A taxonomy of business process modeling and information systems modeling techniques. *International Journal of Flexible Manufacturing Systems*, 13(2):209–228, apr 2001.

[40] Kamal Bhattacharya, Richard Hull, and Jianwen Su. A data-centric design methodology for business processes. In *Handbook of Research on Business Process Modeling, chapter 23*, pages 503–531, 2009.

[41] David Cohn and Richard Hull. Business artifacts: A data-centric approach to modeling business operations and processes. *IEEE Data Eng. Bull.*, 32(3):3–9, 2009.

[42] K. Vergidis, A. Tiwari, and B. Majeed. Business process analysis and optimization: Beyond reengineering. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 38(1):69–82, Jan 2008.

[43] Wil M. P. Van Der Aalst, Arthur H. M. Ter Hofstede, and Mathias Weske. Business process management: A survey. In *Proceedings of*

the 2003 International Conference on Business Process Management, BPM'03, pages 1–12, Berlin, Heidelberg, 2003. Springer-Verlag.

[44] Christodoulos Floudas and Xiaoxia Lin. Mixed integer linear programming in process scheduling: Modeling, algorithms, and applications. *Annals of Operations Research*, 139(1-4):131 – 162, 2005.

[45] Heinrich Rommelfanger. The advantages of fuzzy optimization models in practical use. *Fuzzy Optimization and Decision Making*, 3(4):295–309, December 2004.

[46] Jose M. Pinto and Ignacio E. Grossmann. Assignment and sequencing models for the scheduling of process systems. *Annals of Operations Research*, 81(1-4):433 – 466, 1998.

[47] Carlos A. Mendez, Jaime Cerd, Ignacio E. Grossmann, Iiro Harjunkoski, and Marco Fahl. State-of-the-art review of optimization methods for short-term scheduling of batch processes. *Computers and Chemical Engineering*, 30(6-7):913–946, 2006.

[48] Andreas T. Ernst, Houyuan Jiang, Mohan Krishnamoorthy, and David Sier. Staff scheduling and rostering: A review of applications, methods and models. *European Journal of Operational Research*, 153(1):3–27, 2004.

[49] Chiwoon Cho and Seungsin Lee. A study on process evaluation and selection model for business process management. *Expert Syst. Appl.*,

38(5):6339–6350, May 2011.

[50] Ying Li, Bin Cao, Lida Xu, Jianwei Yin, Shuiguang Deng, Yuyu Yin, and Zhaohui Wu. An efficient recommendation method for improving business process modeling. *Industrial Informatics, IEEE Transactions on*, 10(1):502–513, Feb 2014.

[51] Lila Rao, Gunjan Mansingh, and Kweku-Muata Osei-Bryson. Building ontology based knowledge maps to assist business process re-engineering. *Decis. Support Syst.*, 52(3):577–589, February 2012.

[52] Álvaro Rebuge and Diogo R. Ferreira. Business process analysis in healthcare environments: A methodology based on process mining. *Inf. Syst.*, 37(2):99–116, April 2012.

[53] Ching-Wu Chu, Gin-Shuh Liang, and Chien-Tseng Liao. Controlling inventory by combining {ABC} analysis and fuzzy classification. *Computers & Industrial Engineering*, 55(4):841 – 851, 2008.

[54] Anjali Dhond, Amar Gupta, and Sanjeev Vadhavkar. Data mining techniques for optimizing inventories for electronic commerce. In *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '00, pages 480–486, New York, NY, USA, 2000. ACM.

[55] Chengzhi Jiang and Zhaohan Sheng. Case-based reinforcement learning for dynamic inventory control in a multi-agent supply-chain system.

Expert Systems with Applications, 36(3, Part 2):6520 – 6526, 2009.

[56] Anna-Lena Beutel and Stefan Minner. Safety stock planning under causal demand forecasting. *International Journal of Production Economics*, 140(2):637 – 645, 2012. Sixteenth internationalworkingseminaronproductioneconomics, Innsbruck, 2010.

[57] Ralph D. Snyder, J. Keith Ord, and Adrian Beaumont. Forecasting the intermittent demand for slow-moving inventories: A modelling approach. *International Journal of Forecasting*, 28(2):485 – 496, 2012.

[58] Fotios Petropoulos, Konstantinos Nikolopoulos, Georgios P. Spithourakis, and Vassilios Assimakopoulos. Empirical heuristics for improving intermittent demand forecasting. *Industrial Management & Data Systems*, 113(5):683–696, 2013.

[59] Aris A. Syntetos, M. Zied Babai, and Everette S. Gardner Jr. Forecasting intermittent inventory demands: simple parametric methods vs. bootstrapping. *Journal of Business Research*, 68(8):1746 – 1752, 2015. Special Issue on Simple Versus Complex Forecasting.

[60] Dewayne E. Perry. Enactment control in interact/intermediate. In Brian Warboys, editor, *Software Process Technology, Third European Workshop, EWSPT 94, Villard de Lans, France, February 7-9, 1994, Proceedings*, volume 772 of *Lecture Notes in Computer Science*, pages 107–113. Springer, 1994.

[61] D.E. Perry. Policy-directed coordination and cooperation. In *Software Process Workshop, 1991. Communication and Coordination in the Software Process., Proceedings of the 7th International*, pages 111–113, 1991.

[62] Object Management Group (OMG). Business process modeling language. Technical report, jan 2003.

[63] Yuqun Zhang and Dewayne E. Perry. A goal-directed modeling technique towards business process. In *Service Oriented System Engineering (SOSE), 2014 IEEE 8th International Symposium on*, pages 110–121, April 2014.

[64] Andreas Lanz, Roberto Posenato, Carlo Combi, and Manfred Reichert. Controllability of time-aware processes at run time. In Robert Meersman, Herv Panetto, Tharam S. Dillon, Johann Eder, Zohra Bellahsene, Norbert Ritter, Pieter De Leenheer, and Deijing Dou, editors, *OTM Conferences*, volume 8185 of *Lecture Notes in Computer Science*, pages 39–56. Springer, 2013.

[65] Dewayne E. Perry, Nancy A. Staudenmayer, and Lawrence. Understanding and improving time usage in software development. In Alfonso Fuggetta and Alexander L. Wolf, editors, *Trends in Software Process*, chapter 5. John Wiley and Sons, 1996.

[66] Dewayne E. Perry, Nancy A. Staudenmayer, and Lawrence G. Votta.

Understanding software development: Processes, organisations and technologies. *IEEE software*, 11:36–45, 1994.

[67] Yuqun Zhang and Dewayne E. Perry. A data-centric approach to optimize time in workflow-based business process. In *Services Computing (SCC), 2014 IEEE International Conference on*, pages 709–716, June 2014.

[68] Y. Zhang and D. E. Perry. Generating symbolic business processes in support of evaluating process optimization. In *Services Computing (SCC), 2015 IEEE International Conference on*, pages 754–758, June 2015.

[69] Oracle e-business suite. `http://www.oracle.com/us/products/applications/ebusiness/overview/index.html/`.

[70] John Gurland. Hypothesis testing in time series analysis. *Journal of the American Statistical Association*, 49(265):197–200, 1954.

[71] M. Majidpour, C. Qiu, P. Chu, R. Gadh, and H.R. Pota. Fast prediction for sparse time series: Demand forecast of ev charging stations for cell phone applications. *Industrial Informatics, IEEE Transactions on*, 11(1):242–250, Feb 2015.

[72] Mostafa Majidpour, Charlie Qiu, Peter Chu, Hemanshu R. Pota, and Rajit Gadh. Forecasting the ev charging load based on customer profile or station measurement? *Applied Energy*, 163:134 – 141, 2016.

[73] C. Badica, M. Teodorescu, C. Spahiu, A. Badica, and C. Fox. Integrating role activity diagrams and hybrid idef for business process modeling using mda. In *Symbolic and Numeric Algorithms for Scientific Computing, 2005. SYNASC 2005. Seventh International Symposium on*, pages 4 pp.–, 2005.

[74] Zan Xiao, Donggang Cao, Chao You, and Hong Mei. Towards a constraint-based framework for dynamic business process adaptation. In *Services Computing (SCC), 2011 IEEE International Conference on*, pages 685–692, 2011.

[75] Volker Gruhn and Ralf Laue. Complexity metrics for business process models. In *in: W. Abramowicz, H.C. Mayr (Eds.), 9th International Conference on Business Information Systems (BIS 2006), Lecture Notes in Informatics*, pages 1–12.

[76] Elvira Roln Aguilar, Francisco Ruiz, Flix Garca, and Mario Piattini. Applying software metrics to evaluate business process models. *CLEI Electron. J.*, 9(1), 2006.

[77] Irene T. P. Vanderfeesten, Jorge Cardoso, and Hajo A. Reijers. A weighted coupling metric for business process models. In Johann Eder, Stein L. Tomassen, Andreas L. Opdahl, and Guttorm Sindre, editors, *CAiSE Forum*, volume 247 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2007.

[78] Remco Dijkman, Marlon Dumas, Boudewijn van Dongen, Reina Käärik, and Jan Mendling. Similarity of business process models: Metrics and evaluation. *Inf. Syst.*, 36(2):498–516, April 2011.

[79] Sen Zeng, Shuangxi Huang, and Yushun Fan. Service-oriented business process modeling and performance evaluation based on ahp and simulation. In *e-Business Engineering, 2007. ICEBE 2007. IEEE International Conference on*, pages 469–476, 2007.

[80] Frank Goethals, Manu De Backer, Wilfried Lemahieu, Monique Snoeck, Jacques Vandenbulcke, and SAP-leerstoel Extended Enterprise Infrastructures. Identifying dependencies in business processes. In *Communication and Coordination in Business Processes (LAP-CCBP) Workshop, Kiruna, Sweden, June*, volume 22, 2005.

[81] Georg Grossmann, Michael Schrefl, and Markus Stumptner. Modelling inter-process dependencies with high-level business process modelling languages. In *Proceedings of the fifth Asia-Pacific conference on Conceptual Modelling - Volume 79*, APCCM '08, pages 89–102, Darlinghurst, Australia, Australia, 2008. Australian Computer Society, Inc.

[82] Semih Coskun, Hseyin Basligil, and Hayri Barali. A weakness determination and analysis model for business process improvement. *Business Proc. Manag. Journal*, 14(2):243–261, 2008.

[83] Azim Danesh and Ned Kock. An experimental study of process representation approaches and their impact on perceived modeling quality

and redesign success. *Business Proc. Manag. Journal*, 11(6):724–735, 2005.

[84] Min-Yuan Cheng, Hsing-Chih Tsai, and Yun-Yan Lai. Construction management process reengineering performance measurements. *Automation in Construction*, 18(2):183 – 193, 2009.

[85] Selma Limam Mansar, Hajo A. Reijers, and Fouzia Ounnar. Development of a decision-making strategy to improve the efficiency of {BPR}. *Expert Systems with Applications*, 36(2, Part 2):3248 – 3262, 2009.

[86] Miha kerlavaj, Mojca Indihar temberger, Rok krinjar, and Vlado Dimovski. Organizational learning culturethe missing link between business process change and organizational performance. *International Journal of Production Economics*, 106(2):346 – 367, 2007. Special section on organizational structure, culture and operations management: an empirical missing link.

[87] Michael Freimer, Douglas Thomas, and John Tyworth. The value of setup cost reduction and process improvement for the economic production quantity model with defects. *European Journal of Operational Research*, 173(1):241 – 251, 2006.

[88] Wenan Tan, Weiming Shen, Lida Xu, Bosheng Zhou, and Ling Li. A business process intelligence system for enterprise process performance management. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 38(6):745–756, Nov 2008.

[89] Pavlos Delias, Anastasios D. Doulamis, and Nikolaos F. Matsatsinis. A joint optimization algorithm for dispatching tasks in agent-based workflow management systems. In Joaquim Cordeiro, Jos?and Filipe, editor, *ICEIS (2)*, pages 199–206, 2008.

[90] S. Aiber, Dagan Gilat, A. Landau, N. Razinkov, A. Sela, and S. Wasserkrug. Autonomic self-optimization according to business objectives. In *Autonomic Computing, 2004. Proceedings. International Conference on*, pages 206–213, May 2004.

[91] Soudabeh Khodambashi. Business process re-engineering application in healthcare in a relation to health information systems. *Procedia Technology*, 9(0):949 – 957, 2013. {CENTERIS} 2013 - Conference on {ENTERprise} Information Systems / ProjMAN 2013 - International Conference on Project MANagement/ {HCIST} 2013 - International Conference on Health and Social Care Information Systems and Technologies.

[92] Mohammad Alrifai and Thomas Risse. Combining global optimization with local selection for efficient qos-aware service composition. In *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, pages 881–890, New York, NY, USA, 2009. ACM.

[93] P. Leitner, W. Hummer, and S. Dustdar. Cost-based optimization of service compositions. *Services Computing, IEEE Transactions on*, 6(2):239–251, April 2013.

[94] M.J. Buco, R.N. Chang, L.Z. Luan, E. So, Chunqiang Tang, and C. Ward. Pem: a framework enabling continual optimization of workflow process executions based upon business value metrics. In *Services Computing, 2005 IEEE International Conference on*, volume 2, pages 33–40 vol.2, July 2005.

[95] Tim Weitzel. Process governance and optimization for it reliant business processes: An empirical analysis of financial processes in germany?s fortune 1,000 non-banks. *2014 47th Hawaii International Conference on System Sciences*, 8:196b, 2006.

[96] J.M. Mendes, P. Leita o, F. Restivo, and A.W. Colombo. Process optimization of service-oriented automation devices based on petri nets. In *Industrial Informatics (INDIN), 2010 8th IEEE International Conference on*, pages 274–279, July 2010.

[97] Roger S. Debreceny. Re-engineering it internal controls: Applying capability maturity models to the evaluation of it controls. *2014 47th Hawaii International Conference on System Sciences*, 8:196c, 2006.

[98] Eric Peukert, Henrike Berthold, and Erhard Rahm. Rewrite techniques for performance optimization of schema matching processes. In *Proceedings of the 13th International Conference on Extending Database Technology*, EDBT '10, pages 453–464, New York, NY, USA, 2010. ACM.

[99] Dewei Peng, Liang Cheng, Hongmei Zhou, and Xia Zhang. Study and application of business process optimization and evaluation. In *Services*

Computing Conference (APSCC), 2012 IEEE Asia-Pacific, pages 380–383, Dec 2012.

[100] Hafedh Mili, Guitta Bou Jaoude, Eric Lefebvre, Guy Tremblay, and Alex Petrenko. Business process modeling languages: Sorting through the alphabet soup. In *OF 22 NO. IST-FP6-508794 (PROTOCURE II) SEPTEMBER*, page 2005, 2004.

[101] Alireza Pourshahid, Gunter Mussbacher, Daniel Amyot, and Michael Weiss. An aspect-oriented framework for business process improvement. In Gilbert Babin, Peter Kropf, and Michael Weiss, editors, *E-Technologies: Innovation in an Open World*, volume 26 of *Lecture Notes in Business Information Processing*, pages 290–305. Springer Berlin Heidelberg, 2009.

[102] Massimo Bertolini, M. Bevilacqua, F.E. Ciarapica, and G. Giacchetta. Business process re-engineering in healthcare management: a case study. *Business Process Management Journal*, 17(1):42–66, 2011.

[103] Dusko Ursic, Sandra Anteric, and Matjaz Mulej. Business process re-engineering in practicean example of a medium-sized slovenian company in difficulties. *Systemic Practice and Action Research*, 18(1):89–117, 2005.

[104] Sanjay Goel and Vicki Chen. Can business process reengineering lead to security vulnerabilities: Analyzing the reengineered process. *International Journal of Production Economics*, 115(1):104 – 112, 2008.

[105] Yasin Ozcelik. Do business process reengineering projects payoff? evidence from the united states. *International Journal of Project Management*, 28(1):7 – 13, 2010.

[106] A. Tiwari, K. Vergidis, and B. Majeed. Evolutionary multi-objective optimization of business processes. In *Evolutionary Computation, 2006. CEC 2006. IEEE Congress on*, pages 3091–3097, 2006.

[107] Daniela Grigori, Fabio Casati, Umeshwar Dayal, and Ming-Chien Shan. Improving business process quality through exception understanding, prediction, and prevention. In *Proceedings of the 27th International Conference on Very Large Data Bases*, VLDB '01, pages 159–168, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[108] N. Debnath, C. Salgado, M. Peralta, D. Riesco, and G. Montejano. Optimization of the business process metrics definition according to the bpdm standard and its formal definition in ocl. In *Computer Systems and Applications (AICCSA), 2010 IEEE/ACS International Conference on*, pages 1–8, May 2010.

[109] C.Y. Lam, W.H. Ip, and C.W. Lau. A business process activity model and performance measurement using a time series {ARIMA} intervention analysis. *Expert Systems with Applications*, 36(3, Part 2):6986 – 6994, 2009.

[110] Liang Quan and Guo shuang Tian. A business processes' multi-objective optimization model based on simulation. In *Information Management,*

Innovation Management and Industrial Engineering, 2009 International Conference on, volume 4, pages 572–575, Dec 2009.

[111] W. D. Li and C. A. McMahon. A simulated annealing-based optimization approach for integrated process planning and scheduling. *Int. J. Comput. Integr. Manuf.*, 20(1):80–95, January 2007.

[112] Alexander Nowak, Frank Leymann, David Schumm, and Branimir Wetzstein. An architecture and methodology for a four-phased approach to green business process reengineering. In Dieter Kranzlmller and AMin Toja, editors, *Information and Communication on Technology for the Fight against Global Warming*, volume 6868 of *Lecture Notes in Computer Science*, pages 150–164. Springer Berlin Heidelberg, 2011.

[113] K. Vergidis and A. Tiwari. Business process design and attribute optimization within an evolutionary framework. In *Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence). IEEE Congress on*, pages 668–675, June 2008.

[114] Jay April, Marco Better, Fred Glover, James Kelly, and Manuel Laguna. Enhancing business process management with simulation optimization. In *Proceedings of the 38th Conference on Winter Simulation*, WSC '06, pages 642–649. Winter Simulation Conference, 2006.

[115] Ashutosh Tiwari, Kostas Vergidis, and Rajkumar Roy. Evolutionary optimization of business process designs. In KeshavP. Dahal, KayChen

Tan, and PeterI. Cowling, editors, *Evolutionary Scheduling*, volume 49 of *Studies in Computational Intelligence*, pages 513–541. Springer Berlin Heidelberg, 2007.

[116] MukhammadAndri Setiawan, Shazia Sadiq, and Ryan Kirkman. Facilitating business process improvement through personalized recommendation. In Witold Abramowicz, editor, *Business Information Systems*, volume 87 of *Lecture Notes in Business Information Processing*, pages 136–147. Springer Berlin Heidelberg, 2011.

[117] Claudio Arbib and Fabrizio Marinelli. Integrating process optimization and inventory planning in cutting-stock with skiving option: An optimization model and its application. *European Journal of Operational Research*, 163(3):617 – 630, 2005. Supply Chain Management and Advanced Planning.

[118] Rashmi Jain, Anithashree Chandrasekaran, and Angappa Gunasekaran. Benchmarking the redesign of business process reengineering?curriculum. *Benchmarking: An International Journal*, 17(1):77–94, 2010.

[119] Ned Kock, Jacques Verville, Azim Danesh-Pajou, and Dorrie DeLuca. Communication flow orientation in business process modeling and its effect on redesign success: Results from a field study. *Decision Support Systems*, 46(2):562 – 575, 2009.

[120] Natasa Vujica Herzog, Stefano Tonchia, and Andrej Polajnar. Linkages

between manufacturing strategy, benchmarking, performance measurement and business process reengineering. *Computers and Industrial Engineering*, 57(3):963 – 975, 2009.

[121] Khurram Shahzad and Jelena Zdravkovic. A goaloriented approach for business process improvement using process warehouse data. In Anne Persson and Janis Stirna, editors, *The Practice of Enterprise Modeling*, volume 39 of *Lecture Notes in Business Information Processing*, pages 84–98. Springer Berlin Heidelberg, 2009.

[122] H.A. Reijers and S. Liman Mansar. Best practices in business process redesign: an overview and qualitative evaluation of successful redesign heuristics. *Omega*, 33(4):283 – 306, 2005.

[123] F. Niedermann, S. Radeschutz, and B. Mitschang. Design-time process optimization through optimization patterns and process model matching. In *Commerce and Enterprise Computing (CEC), 2010 IEEE 12th Conference on*, pages 48–55, Nov 2010.

[124] Raphal K. Akamavi. Re-engineering service quality process mapping: e-banking process. *International Journal of Bank Marketing*, 23(1):28–53, 2005.

[125] Feriel Daoudi and Selmin Nurcan. A benchmarking framework for methods to design flexible business processes. *Software Process: Improvement and Practice*, 12(1):51–63, 2007.

[126] K. Shahzad and J. Zdravkovic. A decision-based approach to business process improvement. In *Systems Man and Cybernetics (SMC), 2010 IEEE International Conference on*, pages 810–818, Oct 2010.

[127] Kelly J. Fadel and Mohan Tanniru. A knowledge-centric framework for process redesign. In *Proceedings of the 2005 ACM SIGMIS CPR Conference on Computer Personnel Research*, SIGMIS CPR '05, pages 49–58, New York, NY, USA, 2005. ACM.

[128] Kelly J. Fadel, Susan A. Brown, and Mohan Tanniru. A theoretical framework for knowledge transfer in process redesign. *SIGMIS Database*, 39(3):21–40, July 2008.

[129] Laurence Lock Lee. Balancing business process with business practice for organizational advantage. *Journal of Knowledge Management*, 9(1):29–41, 2005.

[130] Samia M. Siha and Germaine H. Saad. Business process improvement: empirical assessment and extensions. *Business Process Management Journal*, 14(6):778–802, 2008.

[131] Premaratne Samaranayake. Business process integration, automation, and optimization in erp. *Business Process Management Journal*, 15(4):504–526, 2009.

[132] Razvi Doomun and Nevin Vunka Jungum. Business process modelling,

simulation and reengineering: call centres. *Business Process Management Journal*, 14(6):838–848, 2008.

[133] F. Koetter, A. Weisbecker, and T. Renner. Business process optimization in cross-company service networks: Architecture and maturity model. In *SRII Global Conference (SRII), 2012 Annual*, pages 715–724, July 2012.

[134] Four common pitfalls of safety stock. `http://www.opsrules.com/supply-chain-optimization-blog/bid/333778/Four-Common-Pitfalls-of-Safety-St`

[135] Sales forecasting ? dont waste your time. `http://www.sopplan.com/sales-forecasting/sales-forecasting-dont-waste-your-time/`.

[136] Jagan Sankaranarayanan, Hanan Samet, and Amitabh Varshney. A fast k-neighborhood algorithm for large point-clouds. In *Proceedings of the 3rd Eurographics / IEEE VGTC Conference on Point-Based Graphics*, SPBG'06, pages 75–84, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[137] Yan-Ming Zhang, Kaizhu Huang, Guanggang Geng, and Cheng-Lin Liu. Fast knn graph construction with locality sensitive hashing. In Hendrik Blockeel, Kristian Kersting, Siegfried Nijssen, and Zelezn? Filip, editors, *ECML/PKDD (2)*, volume 8189 of *Lecture Notes in Computer Science*, pages 660–674. Springer, 2013.

[138] Kiana Hajebi, Yasin Abbasi-Yadkori, Hossein Shahbazi, and Hong Zhang. Fast approximate nearest-neighbor search with k-nearest neighbor graph. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Two*, IJCAI'11, pages 1312–1317. AAAI Press, 2011.

[139] A. Nico Habermann and Dewayne E. Perry. Well formed system composition. Technical Report CMU-CS-80-117, Department of Computer Science, Carnegie-Mellon University, March 1980.

[140] David Garlan and Mary Shaw. An introduction to software architecture. Technical report, Pittsburgh, PA, USA, 1994.

[141] Ivica Crnkovic. *Building Reliable Component-Based Software Systems.* Artech House, Inc., Norwood, MA, USA, 2002.

[142] Rui C. Gonçalves, Don Batory, and João L. Sobral. Reflo: An interactive tool for pipe-and-filter domain specification and program generation. *Softw. Syst. Model.*, 15(2):377–395, May 2016.

Vita

Yuqun Zhang was born in Ganzhou, Jiangxi Province, China. He received the Bachelor of Engineering degree from the Department of Communications Engineering of Tianjin University in 2008. He then went to the graduate school of the University of Rochester, New York State, USA and worked with Prof. Wendi Heinzelman in the research areas of wireless and mobile networking. He received the Master of Science degree for the Department of Electrical and Computer Engineering of the University of Rochester in 2010. From fall 2010, he began his PhD studies in the Software Engineering track of the Department of Electrical and Computer Engineering of the University of Texas at Austin. He is now working with Prof. Dewayne Perry in the research areas of software engineering and services computing.

Permanent address: yuqun_zhang@utexas.edu

This dissertation was typeset with LaTeX[†] by the author.

[†]LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.