

UNIVERSITY OF MACEDONIA
DEPARTMENT OF APPLIED INFORMATICS
GRADUATE PROGRAM

**Extending Evolutionary Multi-Objective Optimization of
Business Process Designs**

M.Sc. THESIS

of

Konstantinos Georgoulakos

Thessaloniki, February 2019

Extending Evolutionary Multi-Objective Optimization of Business Process Designs

Konstantinos Georgoulakos

B.Sc. in Physics, Physics Department
Aristotle University of Thessaloniki, 2013

M.Sc. Thesis

submitted as a partial fulfillment of the requirements for
THE DEGREE OF MASTER OF SCIENCE IN APPLIED INFORMATICS

Supervisor: Dr Kostas Vergidis

Approved by examining board on 26 February 2019

Prof Nikolaos Samaras

Dr Aggelo Sifaleras

Dr Kostas Vergidis

Abstract

Optimizing a problem to produce a set of improved solutions is not a new concept. Many scientific areas have been benefited by the application of optimizations techniques and so have business processes. The competitive business environments have led organizations into examining and re-designing their core business processes, aiming for improving their performance and market responsiveness. The optimization and the continuous improvement of business processes within a company, can give the advantage to the company to be more competitive by reducing its costs, improving the delivery quality and efficiency, and enabling adaptation to changing environments. This thesis focuses on business process multi-objective optimization with evolutionary algorithms. There have already been optimization approaches with evolutionary algorithms for business process optimization problems that demonstrated rather satisfactory results. This thesis aims to improve and extend those approaches by providing a revised and refined version of an existing business process optimization framework by Vergidis (2008), that incorporates a pre-processing technique for enhancing the efficiency of the employed Evolutionary Multi-objective Optimization Algorithms (EMOAs), a new process composition algorithm that make the new framework capable of fulfilling more real-life constraints and handling more complex problems and many other features such as ease of use, more efficient I/O, better interactivity and easy maintenance. The proposed pre-processing technique was tested as a standalone procedure and demonstrated satisfactory results, managing to reduce drastically the problem dataset of all scenarios examined. The results of the whole optimization framework for the real-life scenarios examined, were very promising and indicated that the framework work as expected. It can automate the process composition and identify alternative business process designs with optimized attribute values.

Conference papers

1. **Kostas Georgoulakos**, George Tsakalidis, Kostas Vergidis and Nikolaos Samaras (2017), 'Evolutionary Multi-objective Optimization of Business Process Designs with Pre-processing', in *Proceedings of IEEE Congress on Evolutionary Computation (CEC'2017)*, San Donostia, Spain, 5-8 June 2017.
2. **Kostas Georgoulakos**, George Tsakalidis and Kostas Vergidis (2017), 'Algorithmic composition of business process designs: Initial experiments with PCA-II', in *Proceedings of IEEE 6th International Symposium & 28th National Conference on Operational Research*, Thessaloniki, Greece, 8-10 June 2017.

Acknowledgements

After an intensive period during attending the M.Sc. in Applied Informatics, I strongly believe that writing this note of thanks is the finishing touch on my thesis. It has been a period of intense learning for me, not only in the scientific arena, but also on a personal level. Writing this thesis has had a big impact on me and I would like to reflect on the ones who have supported and guided me throughout this period.

I would like to thank my supervisor Dr Kostas Vergidis for the opportunity to write this M.Sc. thesis, the continuous guidance and constructive feedback throughout the research.

I would like also to thank my family, close friends and those who supported and encouraged me during the past years.

Table of contents

CHAPTER 1 INTRODUCTION	1
1.1 INTRODUCTION TO BUSINESS PROCESSES	1
1.2 EVOLUTIONARY MULTI-OBJECTIVE OPTIMIZATION.....	3
1.3 PROBLEM STATEMENT	4
1.4 THESIS STRUCTURE	4
1.5 SUMMARY	5
CHAPTER 2 LITERATURE REVIEW	7
2.1 BUSINESS PROCESS DEFINITIONS	8
2.2 BUSINESS PROCESS MODELLING.....	10
2.3 BUSINESS PROCESS OPTIMIZATION (BPO)	29
2.4 SUMMARY	40
CHAPTER 3 AIM & OBJECTIVES.....	41
3.1 RESEARCH AIM	41
3.2 RESEARCH OBJECTIVES	41
3.3 RESEARCH METHODOLOGY	42
3.4 RESEARCH DELIVERABLES	44
3.5 EXPECTED GAINS.....	44
3.6 SUMMARY	46
CHAPTER 4 DEPLOYING PRE-PROCESSING TO EBPO_F	47
4.1 PURPOSE OF DATA PRE-PROCESSING.....	47
4.2 MAIN STEPS OF PRE-PROCESSING	51
4.3 SUMMARY	55
CHAPTER 5 PROCESS COMPOSITION ALGORITHM (PCA-II).....	57
5.1 PROCESS COMPOSITION	57
5.2 NECESSITY OF PCA-II.....	58
5.3 PCA-II	61
5.4 SUMMARY	74
CHAPTER 6 EXTENDED BUSINESS PROCESS OPTIMIZATION FRAMEWORK (EBPO_F)	75
6.1 PROBLEM FORMULATION.....	75
6.2 FRAMEWORK OVERVIEW	77
6.3 FRAMEWORK IMPLEMENTATION.....	89
6.4 SUMMARY	91

CHAPTER 7 TESTING & RESULTS	93
7.1 SCENARIOS.....	93
7.2 VALIDATION OF PRE-PROCESSING.....	96
7.3 VALIDATION OF EBPO _F	99
7.4 SUMMARY	106
CHAPTER 8 DISCUSSION & CONCLUSIONS.....	107
8.1 THESIS OVERVIEW	107
8.2 RESEARCH CONTRIBUTION	109
8.3 RESEARCH LIMITATIONS.....	110
8.4 FUTURE WORK	111
8.5 CONCLUSION	112
REFERENCES.....	113

List of figures

FIGURE 1.1 SCHEMATIC RELATIONSHIP OF THE MAIN BUSINESS PROCESS ELEMENTS BY VERGIDIS (2008).....	2
FIGURE 2.1 BASIC ELEMENTS OF A FLOWCHART BY WWW.SMARTDRAW.COM.....	12
FIGURE 2.2 EXAMPLE OF THE PROCESS OF MEDICAL SERVICES USING A FLOWCHART BY TIRE.DRIVEEASY.CO.....	12
FIGURE 2.3 IDEF0 BASIC SYNTAX BY WWW.IDEF.COM.....	14
FIGURE 2.4 AN EXAMPLE OF “TOP LEVEL CONTEXT DIAGRAM” BY EN.WIKIPEDIA.ORG.....	15
FIGURE 2.5 DECOMPOSITION STRUCTURE OF IDEF0 PROCESS BY EN.WIKIPEDIA.ORG.....	15
FIGURE 2.6 SYMBOLS USED FOR IDEF3 PROCESS DESCRIPTIONS BY EN.WIKIPEDIA.ORG.....	17
FIGURE 2.7 EXAMPLE OF PROCESS FLOW DESCRIPTION BY IT.TOOLBOX.COM.....	18
FIGURE 2.8 EXAMPLE OF OBJECT STATE TRANSITION NETWORK DESCRIPTION BY IT.TOOLBOX.COM.....	18
FIGURE 2.9 BASIC SYNTAX OF USE-CASE DIAGRAMS BY WWW.CONCEPTDRAW.COM.....	20
FIGURE 2.10 BASIC SYNTAX OF ACTIVITY DIAGRAMS BY WWW.CONCEPTDRAW.COM.....	20
FIGURE 2.11 USE-CASE DIAGRAM MODELLING SOME PROCESSES OF THE UNIVERSITY STUDENTS’ SYSTEM.....	20
FIGURE 2.12 ACTIVITY DIAGRAM OF AN EMAIL CONNECTION BY WWW.SMARTDRAW.COM.....	21
FIGURE 2.13 BPMN BASIC ELEMENTS BY ALEXANDER SAMARIN.....	22
FIGURE 2.14 BPD OF AN ON-LINE AUCTION SYSTEM BY WWW.OMG.ORG.....	23
FIGURE 2.15 BPD EXAMPLE WITH POOLS BY WWW.OMG.ORG.....	23
FIGURE 2.16 EXAMPLE OF A POTENTIAL ACTIVITY EQUIVALANCE BY HOFACKER AND VETSCHERA (2001).....	25
FIGURE 2.17 A FEASIBLE BUSINESS PROCESS DESIGN BY HOFACKER AND VETSCHERA (2001).....	26
FIGURE 2.18 A GENERIC BUSINESS PROCESS DESIGN REPRESENTATION BY VERGIDIS (2008).....	28
FIGURE 2.19 REPRESENTAION OF A DESIGN ALTERNATIVE BY HOFACKER AND VETSCHERA (2001).....	31
FIGURE 2.20 EXAMPLE OF TRM BY VERGIDIS (2008).....	33
FIGURE 2.21 TRM MAPPING FOR TASK 1 BY VERGIDIS (2008).....	33
FIGURE 2.22 PSEUDO-CODE FOR CONSTRUCTING THE VISUAL REPRESENTATION PERSPECTIVE BY VERGIDIS 2008.....	33
FIGURE 2.23 BUSINESS PROCESS GRAPH ELABORATION BY VERGIDIS 2008.....	34
FIGURE 2.24 PCA REQUIREMENTS BY VERGIDIS 2008.....	35
FIGURE 2.25 PCA OUTPUTS BY VERGIDIS 2008.....	35
FIGURE 2.26 MAIN STEPS OF PCA BY VERGIDIS 2008.....	37
FIGURE 2.27 ALGORITHM FOR “ELABORATE CHILD LEVEL” PHASE BY VERGIDIS (2008).....	38
FIGURE 2.28 BASIC STEPS FOR THE “UPDATE GRAPH & TRM” OPERATION OF PCA BY VERGIDIS (2008).....	39
FIGURE 3.1 ALTERNATIVE DESIGNS FOR A SPECIFIC SOLUTION.....	45
FIGURE 4.1 PRE-PROCESSING ALGORITHM.....	51
FIGURE 4.2 PSEUDOCODE OF “CHECK GLOBAL INPUTS & OUTPUTS” SUB-PROCESS.....	52
FIGURE 4.3 PSEUDOCODE OF “CHECK TASK INPUTS” SUB-PROCESS.....	52
FIGURE 4.4 PSEUDOCODE OF “CHECK TASK OUTPUTS” SUB-PROCESS.....	53
FIGURE 4.5 PSEUDOCODE OF “CHCEK CATEGORIES” SUB-PROCESS.....	54

FIGURE 4.6 PSEUDOCODE OF “CHECK GLOBAL AVAILABILITY”	55
FIGURE 5.1 EVALUATION PHASE.....	57
FIGURE 5.2 SCENARIO B WITH AND WITHOUT PRE-PROCESSING FOR BPO _F	58
FIGURE 5.3 SCENARIO B BY VERGIDIS (2008).....	60
FIGURE 5.4 A 4-TASK OPTIMIZED PROCESS DESIGN BY VERGIDIS (2008)	60
FIGURE 5.5 MAIN STEPS OF PCA BY VERGIDIS (2008)	61
FIGURE 5.6 “ELABORATE CHILD LEVEL” OPERATION BY VERGIDIS (2008)	61
FIGURE 5.7 A FEASIBLE PROCESS DESIGN	62
FIGURE 5.8 COMPOSITION APPROACH OF PCA-II	63
FIGURE 5.9 COMPOSITION OUTCOMES OF PCA-II.....	65
FIGURE 5.10 SAME TASKS IN DIFFERENT ORDER.....	66
FIGURE 5.11 TASKS IN DIFFERENT ORDER	66
FIGURE 5.12 CALCULATION OF DOI	68
FIGURE 5.13 MAIN STEPS OF PCA-II	69
FIGURE 5.14 PSEUDOCODE OF PCA-II	70
FIGURE 5.15 ALGORITHM OF “ATTACH TASKS” OPERATION.....	71
FIGURE 5.16 TWO FESIBLE DESIGNS ELABORATED WITH GREEATY MODE ENABLED	72
FIGURE 5.17 ALGORITHM OF “ASSESS CONTAINER” OPERATION.....	73
FIGURE 5.18 ALOGORITHM OF “HOLD THE BEST PARTIALLY CONNECTED GRAPH” OPERATION	74
FIGURE 6.1 THE MAIN COMPONENTS OF THE PROPOSED OPTIMIZATION FRAMEWORK NY VERGIDIS (2008)	77
FIGURE 6.2 CONFIGURATION FILE OF EBPO _F	78
FIGURE 6.3 PART OF SCENARIOC CSV FILE	79
FIGURE 6.4 CONFIGURATION FILE FOR NSGA ₂	79
FIGURE 6.5 OUTCOMES OF EBPO _F EXECUTION.....	80
FIGURE 6.6 MAIN STEPS OF EBPO _F	82
FIGURE 6.7 THE DICTIONARY WITH THE PROBLEM SPECIFICATION	83
FIGURE 6.8 THE CROSSOVER OPERATOR.....	85
FIGURE 6.9 THE MUTATION OPERATOR	86
FIGURE 6.10 THE CONTAINER WITHIN EBPO _F	89
FIGURE 6.11 THE LIBRARIES USED IN EBPO _F	90
FIGURE 6.12 SCREENSHOT OF THE PYTHON PROGRAMMING ENVIRONMENT	91
FIGURE 7.1 BUSINESS PROCESS DESIGN DRAFT FOR ONLINE ORDER PLACEMENT BY VERGIDIS (2008)	94
FIGURE 7.2 BUSINESS PROCESS DESIGN DRAFT FOR SALES FORECASTING BY VERGIDIS (2008).....	95
FIGURE 7.3 BUSINESS PROCESS DESIGN DRAFT FOR FRAUD INVESTIGATION BY VERGIDIS (2008)	96
FIGURE 7.4 LIBRARY OF TASKS AND RESOURCES FOR ONLINE ORDER PLACEMENT BY VERGIDIS (2008).....	97
FIGURE 7.5 LIBRARY OF TASKS AND RESOURCES FOR SALES FORECASTING BY VERGIDIS (2008)	97
FIGURE 7.6 LIBRARY OF TASKS AND RESOURCES FOR FRAUD INVESTIGATION BY VERGIDIS (2008).....	98

FIGURE 7.7 CONFIGURATION OF EBPO _F FOR VALIDATION TESTING	99
FIGURE 7.8 SCENARIO A UNDER NSGA2	99
FIGURE 7.9 SCENARIO A UNDER SPEA2	100
FIGURE 7.10 SCENARIO A UNDER DCD	100
FIGURE 7.11 OPTIMIZED BUSINESS PROCESS DESIGNS FOR SCENARIO A.....	101
FIGURE 7.12 THE OPTIMAL 3-TASK PROCESS DESIGN FOR SCENARIO A.....	101
FIGURE 7.13 SCENARIO B UNDER NSGA2	102
FIGURE 7.14 SCENARIO B UNDER SPEA2	102
FIGURE 7.15 SCENARIO B UNDER DCD	103
FIGURE 7.16 OPTIMIZED BUSINESS PROCESS DESIGNS FOR SCENARIO B.....	103
FIGURE 7.17 SCENARIO C UNDER NSGA2	104
FIGURE 7.18 SCENARIO C UNDER SPEA2	104
FIGURE 7.19 SCENARIO C UNDER DCD	105
FIGURE 7.20 OPTIMIZED BUSINESS PROCESS DESIGNS FOR SCENARIO C.....	105

List of tables

TABLE 2.1 BUSINESS PROCESS DEFINITIONS EXISTING IN LITERATURE	9
TABLE 2.2 SCOPE OF IDEF METHODS BY EN.WIKIPEDIA.ORG	13
TABLE 2.3 TYPES OF DIAGRAMS SUPPORTED BY UML BY WWW.UML.ORG	19
TABLE 2.4 BASIC ELEMENT CATEGORIES OF BPMN BY EN.WIKIPEDIA.ORG	22
TABLE 6.1 PARAMETERS FOR BUSINESS PROCESS PROCESS OPTIMIZATION PROBLEM.....	75

CHAPTER 1

Introduction

In modern times, business environments are constantly changing at a very fast pace. Therefore, organizations should be able to quickly adapt to the new changes in order to hold their market position and advance with the passage of time. The adaptation often involves the re-designing of their core business processes and aims for improving the business performance and the market responsiveness. The key factor for companies to effectively compete in today's volatile business environment, is the effective designing and management of business processes. The optimization and the continuous improvement of business processes within a company, can give the advantage to the company to be more competitive by reducing its costs, improving the delivery quality and efficiency, and enabling adaptation to changing environments.

This research focuses on business process multi-objective optimization with evolutionary algorithms. It is heavily based on a previous approach by Vergidis (2008) where a new framework was introduced for business process optimization using the most state-of-the-art genetic algorithms. This thesis aims to improve the performance of that framework and make it capable of fulfilling more real-life constraints and handling more complex problems. This chapter introduces the concepts of business processes and evolutionary computing. In addition, the problem statement is provided, and the chapter concludes with the structure of this thesis.

1.1 Introduction to business processes

This section introduces the main concept of this research which is the business process. Business processes are omnipresent at the core of almost any organization and reveal the behavior and the workings of an organization. The success of an organization usually derives from the effective usage and stream of the resources within the organization to shape the end product, hence it doesn't suffice to have a solid net capital or possess valuable knowledge, yet, it is essential to be able to behave in accordance with the external environment.

However, the term "business process" is too generic and vague to describe the functionality, the needs and the requirements of all disciplines. Gunasekaran and Kobu (2002) perceive the business process as a group of related tasks that combined, create value for a customer. On the other hand, Castellanos *et al.* (2004) use the term "business" process to denote a set of activities that together achieve a certain business goal. Since the 1990's when the first definitions appeared, there is no

common perception of what a business process is, and several authors introduced their own version of business process definition which was oriented towards a particular direction highlighting only specific aspects. Although most definitions are similar in terms of the concepts used to express and describe business processes, they have received criticisms for not adequately highlighting the business context and not sufficiently distinguishing from manufacturing or production processes. Volkner and Werners (2000) support that no generally accepted definition of the term “business process” exists because of the different disciplines that have approached business processes. Consequently, the main issue with the business process definitions is attributed either to their simplistic and generic nature or to their specific application area. In this research, the author adopts the business process definition by Vergidis (2008) where he perceives a business process as a collective set of tasks that when properly connected and sequences perform a business operation. The aim of the business process is to perform a business operation.

Although there are several different business process definitions in literature, when it comes to the structure of business processes there is a common way of seeing the participating elements. Vergidis (2008) presented a hierarchical schema for business processes involving the most common structural elements found in literature. The solid arrows correspond to the main elements of the schema while the dashed arrows denote the optional elements.

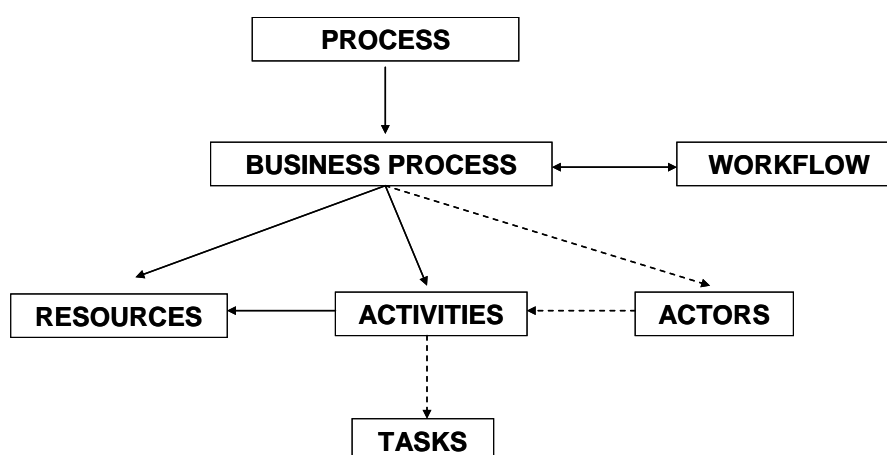


Figure 1.1 Schematic relationship of the main business process elements by Vergidis (2008)

Business processes are a sub-class of generic processes; thus, they inherit all their main characteristics such as resources and activities. The workflow is placed in parallel with the business processes because many times they are interchangeable. If you have the workflow, you can shape the business process and the opposite. In addition, the resources, the activities and the actors constitute the basic structural elements in most business process definitions. Actors are sometimes involved in a business process definition (Lindsay *et al.*, 2003) or sometimes perceived as external entities that enact or execute the process. Activities are widely accepted as the central elements that execute the basic business process steps utilizing the process inputs in order to produce the

desired outputs. The resources are frequently classified as inputs or input resources and as outputs or output resources. The inputs are necessary for the activities to be executed and the outputs result from their execution. Finally, the tasks are perceived as the smallest analyzable element of a business process (Orman, 1995). However, they are usually overlooked by most authors or tend to be another synonym for activities.

1.2 Evolutionary multi-objective optimization

The nature of the real-world problems usually entails the accomplishment of multiple objectives. A multi-objective optimization problem involves minimization and/or maximization and is subjected to a number of constraints. The business process optimization problem is inherently a multi-objective optimization problem due to the variety of factors that a business process can be evaluated with. As the title of this thesis reveals, the adopted optimization approach in this research is based on the evolutionary computing.

In the natural world, evolution has created an unimaginably diverse range of designs, having much greater complexity than mankind could ever hope to achieve. Evolutionary algorithms (EA) mimic nature's evolutionary principles to guide the optimization process towards discovering optimal solutions and they have already been successfully applied to several combinatorial problems. They progress iteratively by growing or developing a population of solutions. This population is then selected in a guided random search using parallel processing to achieve the desired end. Such processes are often inspired by biological mechanisms of evolution.

Initially, a random population of solutions is generated which constitutes the first generation. In addition, a fitness function is involved in the performance evaluation of those solutions and facilitates the selection of the parents for the next generation. This selection is biased towards solutions with higher fitness values. The reproduction of the parents is achieved by the application of operators such as crossover and/or mutation. The crossover operator acts on two selected parents and results in one or two children. The mutation operator acts on an individual solution and results in a new one. These operators create the offspring population of solutions. This process is repeated until a population of solutions of high quality is found, or a previously defined number of generations is reached.

One of the main advantages of EC, is that in each iteration, a population of solution is found instead of a single solution. This enables them to identify a number of optimal solutions in the final population. Another advantage of EC is also the lack of preference towards a specific optimization objective which gives them the capability of providing a wide range of optimal solutions that each of them reflects a different trade-off among the optimization objectives. Consequently, both these

advantages make EC very promising for optimizing business process designs for three main reasons: (1) business process designs that would otherwise be overlooked by a human designer can be discovered by evolutionary algorithms, (2) evolving a solution over the generations can transform an infeasible process design to a feasible one and (3) based on specific objectives, the fittest process design can be determined by evaluating a significant number of alternative designs based on the same process.

1.3 Problem statement

The benefits of EC have also been recognized by Vergidis (2008) and led him to propose a novel evolutionary optimization framework for business processes, BPO_F. This framework used a specific business process representation technique, a process composition algorithm (PCA) and a series of evolutionary algorithms in order to generate optimized business process designs. That framework forms the basis for this research where the author attempts to improve that framework in terms of performance and extend it in handling more complex problems. The outcome of this research is a revised and improved version of that framework, eBPO_F. The new framework incorporates a pre-processing technique for enhancing the efficiency of the employed Evolutionary Multi-objective Optimization Algorithms (EMOAs), a new process composition algorithm, PCA-II, suitable for real-world problems and many other features such as ease of use, more efficient I/O, better interactivity and easy maintenance.

1.4 Thesis structure

The rest of the thesis structure is built in a way that follows the steps within this research and unfolds all the aspects of the new framework.

In more detail:

- ❑ The following chapter, **chapter 2** concerns the literature review and discusses the main concepts of business process optimization. The main subjects are the definition of business processes, the most popular modelling techniques and the most comprehensive evolutionary multi-objective business process optimization approaches found in literature.
- ❑ **Chapter 3** specifies the aim and objectives of this thesis. Having studied and understood the previous approaches for business process optimization, this chapter introduces the goal and the methodology to accomplish it.
- ❑ **Chapter 4** introduces the first goal of this thesis which is a pre-processing algorithm for business process optimization problems aiming for improving the efficiency of the evolutionary algorithms so as for them to produce better solutions.

- ❑ **Chapter 5** introduces the new process composition algorithm. This algorithm follows a different approach from its predecessor and enables the new framework to handle more complex problems.
- ❑ **Chapter 6** introduces the extended business process optimization framework. The tool has been developed in Python and forms the revised and improved version of an existing optimization framework towards usability, I/O, interactivity and maintenance.
- ❑ **Chapter 7** presents the results of the validation testing of the new framework. First, the testing of the pre-processing stage is performed as a standalone application and then the whole framework is executed for each of the EMOAs employed to validate the proper functionality of the new process composition algorithm. A small discussion about the performance of the new framework for each scenario examined, is also provided.
- ❑ **Chapter 8** holds the conclusions extracted from the development of the new business process optimization framework. It also discusses about the contributions of this research, its limitations and provides the author's suggestions for future work.

1.5 Summary

This chapter provided a generic and short discussion about the topics of business processes and evolutionary multi-objective optimization. The problem statement revealed the scope of this thesis and the thesis structure shortly presented the context of each of the following chapters. The next chapter provides the literature review survey on the subjects discussed in the current chapter, in order to follow the remaining part of this thesis which is the development and improvement of a new framework for business process optimization problems.

CHAPTER 2

Literature review

This chapter discusses the main concepts around business process optimization deriving from the problem statement discussed in the previous chapter. The literature survey within this research focuses on the aspects of definition, modelling and evolutionary optimization of business processes. In this chapter an overview of the existing techniques and approaches is provided, in order to highlight their strengths and weaknesses. Furthermore, the overview of these approaches facilitates in the identification of the assumptions that should be considered for the development of the new optimization framework.

The next section provides an overview of the most common definitions found in literature for business processes. There are several different definitions in literature deriving from different areas and thus, there is no common perception on what a business process exactly is. Therefore, the next section attempts to clarify how business processes are perceived, by presenting the most representative ones.

In the second section of the literature review, the main business process modelling techniques are presented. Modelling techniques have been popular and applicable in many problems, from the object-oriented programming to shop-floor activities within industry. Moving to business processes, a process model provides a clear and comprehensive representation of the process in order to extract valuable performance measures and gain profound knowledge of them. Furthermore, Aguilar-Saven (2004) claims that business process modelling establishes a common understanding and analysis of a business process and enables enterprises to be analyzed and integrated through their processes.

The final section presents the two most comprehensive approaches found in literature for evolutionary multi-objective business process optimization. The development of the new optimization framework is heavily based on those two approaches and thus, a detailed discussion is provided for the used business process representation, the assumptions of a feasible process design and the mechanism that enables evolutionary algorithms to be applied to business process optimization problems.

2.1 Business Process Definitions

This section introduces the various business process definitions existing in literature. The reason behind such diversity is that every author describes a business process model highlighting only specific aspects based on the field of study he comes from. It is worth mentioning that there is not such a definition globally accepted and none of the existing definitions prevails over the others. As Shen *et al.* (2004) stating, each business process definition attempt has its own advantages and disadvantages but what remains the same is that each method is used to represent a certain view of enterprise. The aim of this section is to provide an insight towards the main concepts around business processes and clarify how these are perceived by the authors.

The first definitions of business processes appeared in literature in the 1990's and almost any of them seems to be an improved version of the business process definitions provided by Hammer and Champy (1993) and Davenport (1993). Lindsay (1993), Melao and Pidd (2000) and Tinnila (1995) have gathered such definitions which are provided in table 2.1. This table depicts the diversity of the existing business process definitions in literature.

Author(s)	Business process definitions
Agerfalk (1999)	A <i>business process</i> consists of activities ordered in a structured way with the purpose of providing valuable results to the customer.
Castellanos <i>et al.</i> (2004)	The term <i>business process</i> is used to denote a set of activities that collectively achieve a certain business goal. Examples of these processes are the hiring of a new employee or the processing of an order.
Davenport and Short (1990)	<i>Business process</i> is a set of logically related tasks performed to achieve a defined business outcome.
Davenport (1993)	<i>Business process</i> is defined as the chain of activities whose final aim is the production of a specific output for a particular customer or market
Fan (2001) Shen <i>et al.</i> (2004)	<i>Business process</i> is a set of one or more linked procedures or activities that collectively realise a business objective or policy goal, normally within the context of an organisational structure defining functional roles and relationships.
Gunasekaran and Kobu (2002)	A group of related tasks that together create value for a customer is called a <i>business process</i> .
Hammer and Champy (1993)	A <i>business process</i> is a collection of activities that takes one or more kinds of inputs and creates an output that is of value to the customer. A business process has a goal and is affected by events occurring in the external world or in other processes.
Irani <i>et al.</i> (2002)	A <i>business process</i> is a dynamic ordering of work activities across time and place, with a beginning, an end, and clearly identified inputs and outputs.

Johanson <i>et al.</i> (1993)	A business process is a set of linked activities that takes an input and it transforms it to create an output. It should add value to the input and create an output that is more useful and effective to the recipient.
Pall (1987)	Business process is the logical organisation of people, materials, energy, equipment and procedures into work activities designed to produce a specified end result.
Soliman (1998)	<i>Business process</i> may be considered as a complex network of activities connected together.
Stock and Lambert (2001)	A <i>business process</i> can be viewed as a structure of activities designed for action with focus on the end customer and the dynamic management of flows involving products, information, cash, knowledge and ideas.
Stohr and Zhao (2001)	A <i>business process</i> consists of a sequence of activities. It has distinct inputs and outputs and serves a meaningful purpose within an organisation or between organisations.
Volkner and Werners (2000)	<i>Business process</i> is defined as a sequence of states, which result from the execution of activities in organisations to reach a certain objective.
Wang and Wang (2005)	<i>Business process</i> is defined as a set of business rules that control tasks through explicit representation of process knowledge.
Vergidis (2008)	<i>Business process</i> is a collective set of tasks that when properly connected and sequenced perform a business operation. The aim of a business process is to perform a business operation, i.e. any service-related operation that produces value to the organization.

Table 2.1 Business process definitions existing in literature

As it seems from table 2.1 and stated before, most definitions are somewhat related to those by Davenport (1993) and Hammer and Champy (1993). The differences found among them, rely on the emphasis than the authors give to specific aspects of business processes and all of them except Vergidis (2008), have received criticisms for not sufficiently identifying the business component and not clearly distinguishing them to manufacturing or production processes. Agerfalk (1999), Davenport (1993), Hammer and Champy (1993), Stock and Lambert (2001) and Gunasekaran and Kobu (2002), provide more customer oriented definitions. Castellanos *et al.* (2004), Fan (2001) and Shen *et al.* (2004) emphasize on the goal orientation of a business process. Agerfalk (1999) sees business processes as an ordered structure of activities. Pall (1987) who has provided one of the earliest definitions, also involves the human factor in the context of business processes along with the material resources and sees business processes as a structure of all of them logically connected. The term of logical connection is also referred in the definition provided by Davenport and Short (1990) and the term of proper connection and sequence, which is similar, by Vergidis (2008). On the other hand, Soliman (1998) identifies the complexity that a business process may have through his definition. Stock and Lambert (2001) and Irani *et al.* (2002) point out the necessity of clearly identified inputs and outputs for a business process. Hammer and Champy (1993) highlight the fact that a business process may be affected by the external world or the execution of other processes. Additionally, Stock and Lambert (2001) imply through their definition, that the

management of activities and resources participating in a business process may alter dynamically. Furthermore, there are also two definitions worth mentioning, coming from Volkner and Werners (2000) and Wang and Wang (2005) respectively. The first one, emphasizes on states as the main structural elements of a business process. This attempt provides a different insight into business processes as evolving series of states that modify the result of the execution of the participating activities. The second one, introduces business processes as a set of rules that control tasks; unfortunately, without mentioning who is in charge for executing these tasks and if they have an ordered structure. Finally, from the author's point of view, the definition coming from Vergidis (2008) is the most comprehensive definition found in literature since he sees business processes as a collective set of tasks properly connected and sequenced and goes beyond others by approaching the aim of business processes as the fulfilment of a business operation. This approach encompasses all types of business processes without giving emphasis on any specific aspects.

2.2 Business Process Modelling

Business process modeling (BPM) in business process management and systems engineering is the activity of representing processes of an enterprise, so that the current process may be analyzed, improved and automated. The context of business process modelling indicates and facilitates the level of perception and understanding of business processes within a company. As human beings, we can process and understand things better if we can see them. Therefore, the elements and the capabilities of a business process model play a significant role in the business world. According to Lutthuis *et al.* (2001) a main objective of business process intelligence is to provide an insight in the structure of business processes and the relation among them. This insight can be easily obtained by creating business process models that clearly and precisely illustrate the essence of the business organization. These models should contain organizational level details, capabilities for easily identifying bottlenecks and quick assessment of the consequences of a potential change to the customers and the organization itself. According to van der Aalst *et al.* (2003), business process modelling is used to characterize the identification and specification of business processes. Business process modelling includes modelling of activities and their causal and temporal relationships as well as specific business rules that process activities must comply with. Lindsay *et al.* (2003) describe business process modelling as a snapshot of what is perceived at a point of time regarding the actual business process. The objective of business process modelling, as provided by Sadiq and Orłowska (2000), is the high-level specification of processes, while Biazzo (2002) says that it is the representation of relationships between the activities, people, data and objects involved in the production of a specified output. Volkner and Werners (2000) and Aguilar-Saven (2004) claim that business process modelling is essential for the analysis, evaluation and improvement of business processes as it is used to structure the process, such that the existing

and alternative sequence of tasks can be analyzed systematically and comprehensively. As Guha *et al.* (1993) and Abate *et al.* (2002) state, business process modelling is a useful tool to capture, structure and formalize the knowledge about business processes. Aguilar-Saven (2004) suggest that business process models are mainly used to learn about the process, to make decisions on the process, or to develop business process software. Shen *et al.* (2004) supports that business process modelling is an essential part of developing an enterprise information system. According to Vergidis (2008), the business process design is the representation of a business process depicting the participating tasks and their connectivity patterns that determine the flow of the process. The aim of such a design, is to capture, visualize and communicate a business process.

In this section, the author provides an overview of the most significant business process modelling techniques existing in literature. The necessity behind an overview like this, is because business process models are mainly used either to learn about the business process itself, as stated before, or to make decisions on the process or to develop business process software. As it is evident, such purposes involve an extension over some model characteristics. Considering these characteristics, the main modelling concepts can be classified into two major groups. The first classification can be formed by the modelling techniques using a visual diagram, called as diagrammatic models. On the other hand, the second classification corresponds to models consisting of elements that have a mathematical or a formal fundament. Both classifications are presented below along with the most representative examples of each of them.

2.2.1 Diagrammatic Models

The first and most straightforward business process modelling techniques were plain graphical representations and were initially developed for software specification ((Knuth, 1963), (Chapin N., 1971)). The main characteristic of such techniques is the common approach to depict a business process by using a diagram with defined notation e.g. shapes, lines, arrows etc. These diagrammatic techniques have the prominent advantage of illustrating the business process, hence making it easy to follow and understand without the need of any technical expertise. However, if there is no universal standard notation and methodology used, this can lead to misunderstandings about a business process model among people (Havey, 2005). For this reason, BPMN which stands for Business Process Model and Notation, has been developed and is mainly used nowadays among businesses. BPMN will be further discussed later in this section.

2.2.1.1 Flowchart technique

The Flowchart model is probably the first and most popular process notation since it has frequently been used over many years to represent algorithms, workflows and processes.

Flowchart is defined by Lakin *et al.* (1996) as a formalized graphic representation of a program logic sequence, work or manufacturing process, organization chart or similar formalized structure. Flowcharts consist of special symbols representing different types of actions or steps in a process, along with lines and arrows indicating the sequence of steps, and the relationships among them. The basic symbols of a Flowchart are represented in figure 2.1.






Symbol	Name	Function
	Start/end	An oval represents a start or end point.
	Arrows	A line is a connector that shows relationships between the representative shapes.
	Input/Output	A parallelogram represents input or output.
	Process	A rectangle represents a process.
	Decision	A diamond indicates a decision.

Figure 2.1 Basic elements of a flowchart by www.smartdraw.com

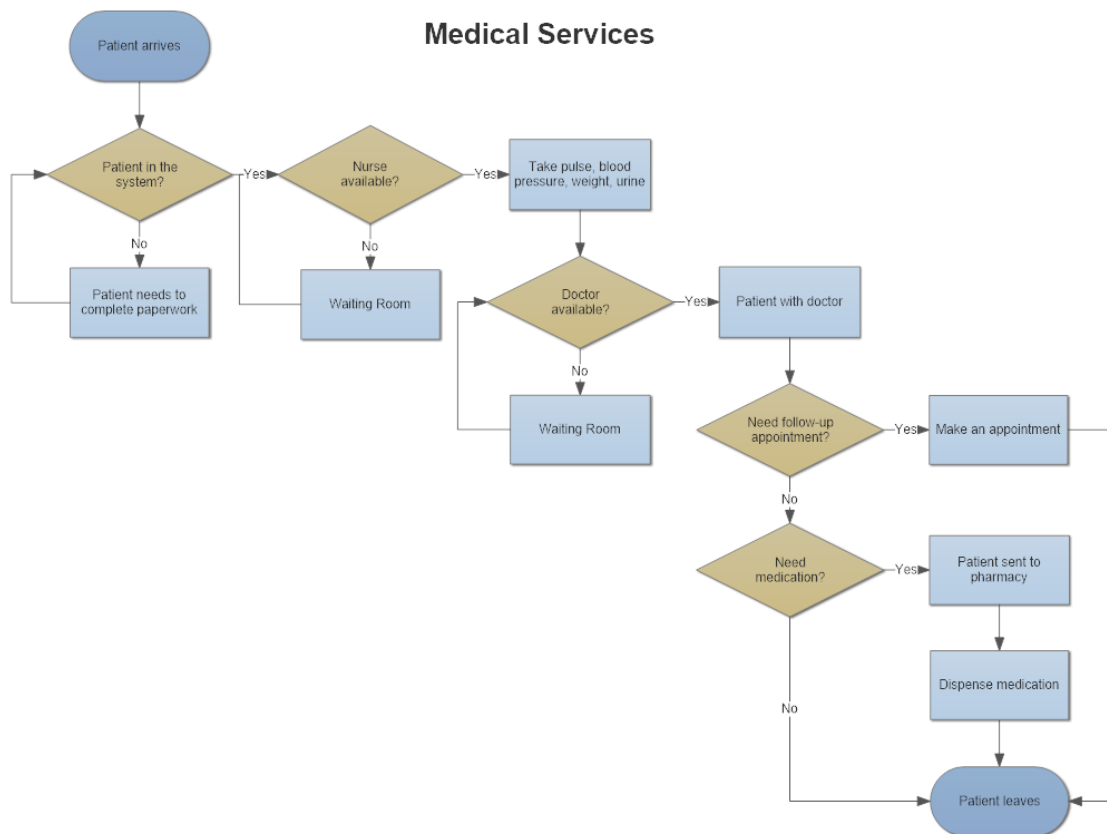


Figure 2.2 Example of the process of medical services using a flowchart by tire.driveeasy.co

Figure 2.2 represents an example of the flowchart technique using the symbols presented in figure 2.1 and demonstrates the simplicity of this technique. The main advantages of this method are the very easy follow-up of the described process, the quick and easy drawing, the flexibility it provides, as a process can be described in various ways, and the communication ability provided by the

standard notation. For example, the process described in figure 2.2 can be easily understood as a medical service process, despite the label, and it seems that no special effort made to draw it.

On the contrary, flowcharts may become too large in effort to capture more and more information within it, hence more difficult to be read. In addition, most of the times, the flexibility comes with no standard methodology and the boundaries of a business process may become unclear. For example, someone could draw the models of all sub-processes of figure 2.2 within the same model, making it very large and difficult to read. Someone could also draw another model for the process of figure 2.2 by setting another step between nurse availability and doctor availability to check for doctor availability of other specialty to take pulse, blood pressure, weight and urine.

To sum up, the best use of the flowchart model technique is for the high-level understanding of a business process and if someone needs to provide much information and many details about a business process, he must choose another modelling technique.

2.2.1.2 Integrated Definition for Function Modelling (IDEF)

The lack of a standard methodology and necessary semantics to support more complex and standardized structures in the flowchart technique, led to the development of standard methodologies such as IDEF and Unified Modelling Language (UML) for process modelling and/or software development. In this section, we are going to discuss IDEF and we will see UML in another section.

IDEF is a family of modelling languages in the field of systems and software engineering, capable of graphically representing a wide range of business, manufacturing and other types of enterprise operations to any level of detail. According to Kim *et al.* (2003), IDEF provides a suite of graphical modelling techniques designed to specify and communicate important aspects of business processes. IDEF was initially developed by US Air Force Materials Laboratory in the mid-1970s as a part of the Integrated Computer-Aided Manufacturing (ICAM). The ICAM program office deemed it valuable to create a “neutral” way of describing the data content of large-scale systems and proceeded with developing methods for processing data independently of the way it was physically stored. The IDEF methods are classified according to the applications they are used. Table 2.2 shows the scope of each method

Method	Scope
IDEF0	Function modelling
IDEF1	Information modelling
IDEF1x	Data modelling
IDEF2	Simulation model design
IDEF3	Process description capture
IDEF4	Object-oriented design
IDEF5	Ontology description capture
IDEF6	Design rationale capture
IDEF7	Information system auditing
IDEF8	User interface modelling
IDEF9	Business constraint discovery
IDEF10	Implementation architecture modelling
IDEF11	Information artefact modelling
IDEF12	Organization modelling
IDEF13	Three schema mapping design
IDEF14	Network design

Table 2.2 Scope of IDEF methods by en.wikipedia.org

of IDEF family. Considering the scope of this research and the table 2.2, the author is going to discuss the IDEF0 and IDEF3 methods since these are related to process modelling.

IDEF0 is a functional modelling method designed to model the decisions, actions and activities within an organization or system. It is used for analyzing, communicating and understanding the functional perspective of a system and the relationships within it. For example, where a flowchart model is used to show the functional flow of a process, IDEF0 is used to show data flow, system control, and the functional flow of lifecycle processes. IDEF0 models consist of a hierarchical series of diagrams, text and glossary cross-referenced to each other. The two primary modelling components are the functions, represented by boxes, and the data and objects that inter-connect those functions, represented by arrows. The basic syntax for an IDEF0 model is shown below in the figure 2.3.

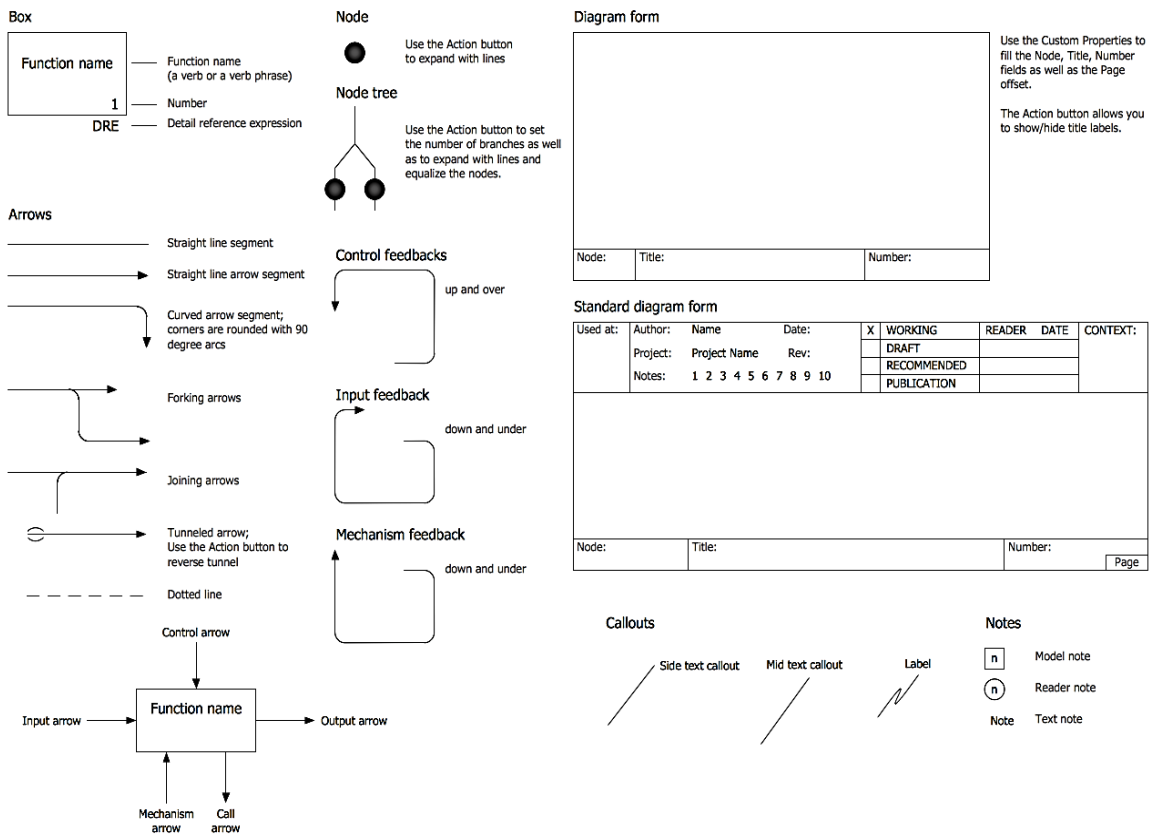


Figure 2.3 IDEF0 basic syntax by www.idf.com

An IDEF0 process starts with the identification of the prime function to be decomposed. This function is identified on a “Top Level Context Diagram” that defines the scope of a particular IDEF0 analysis. An example of a “Top Level Context Diagram” is illustrated in figure 2.4.

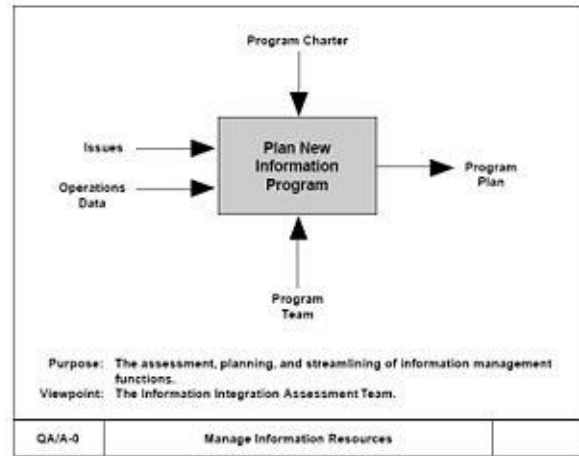


Figure 2.4

An example of “Top Level Context Diagram” by en.wikipedia.org

Then, the prime function can be logically decomposed into its component functions. This process can be continued recursively to the desired level of detail. An example of the IDEF0 process decomposition is presented in figure 2.5 below.

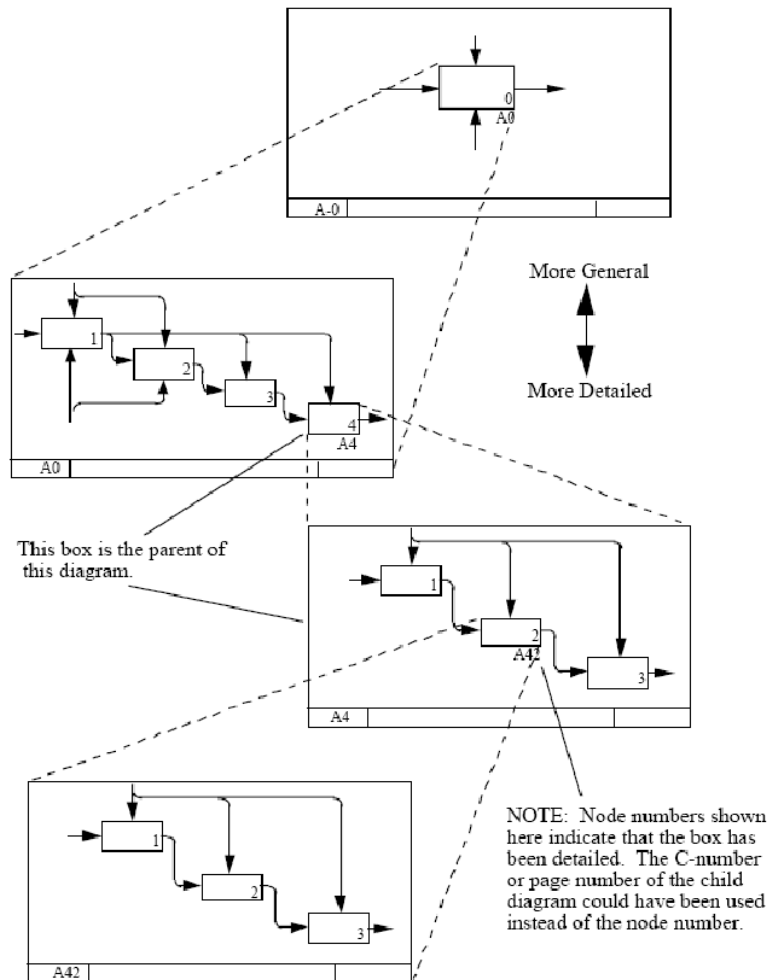


Figure 2.5 Decomposition structure of IDEF0 process by en.wikipedia.org

One of the strengths of IDEF0 modelling technique is the extended level of detail that can be provided, making the model as descriptive as necessary for the decision-making task to be at hand. Additionally, another strength of IDEF0 emerges from its hierarchical nature by facilitating the development of (AS-IS) models that have a top-down representation and interpretation, but which are based on a bottom-up analysis process.

On the other hand, one potential disadvantage comes from the level of detail described in an IDEF0 model and if it is very concise, it may be understandable only from readers who are domain experts or have participated in the model development. In addition, another weakness is the tendency of IDEF0 models to be interpreted as representing a sequence of activities even though IDEF0 is not intended to be used for modeling activity sequences. The activities may be placed in a left to right sequence within a decomposition and connected with the flows. It is natural to order the activities left to right because, if one activity outputs a concept that is used as input by another activity, drawing the activity boxes and concept connections is clearer. The solution to this weakness has been given by IDEF3 which is described below.

IDEF3 is a process description capture method to capture descriptions of sequences of activities, which is considered the common mechanism to describe a situation or process. It is a business process modelling method complementary to IDEF0. The difference between IDEF0 and IDEF3 is that the former shows what is done within an organization or system while the latter shows how things work with it. IDEF3 provides a mechanism for collecting and documenting processes. It captures the precedence and causality relations between situations and events in a form natural to domain experts by providing a structured method for expressing knowledge about how a system, process, or organization works. The basic organizing structure for IDEF3 process descriptions is the notion of scenario. A scenario can be thought as a recurring situation, or a set of situations that describe a typical class of problems addressed by an organization or system, or the setting within which a process occurs. Scenarios establish the focus and boundary conditions of a description and humans must describe what they know in terms of an ordered sequence of activities within the context of the given scenario or situation.

IDEF3 provides two description modes: The Process Flow Description which captures the knowledge of “how things work” in an organization or system and the Object State Transition Network Description which summarizes the allowable transitions of an object throughout a particular process. Both the Process Flow Description and Object State Transition Network Description contain units of information that make up the system description. These model entities, as they are called, form the basic units of an IDEF3 description. The resulting diagrams and text comprise what is termed a “description” as opposed to the focus of what is produced by

the other IDEF methods whose product is a “model.” The basic syntax for an IDEF3 process description is shown below in the figure 2.6.

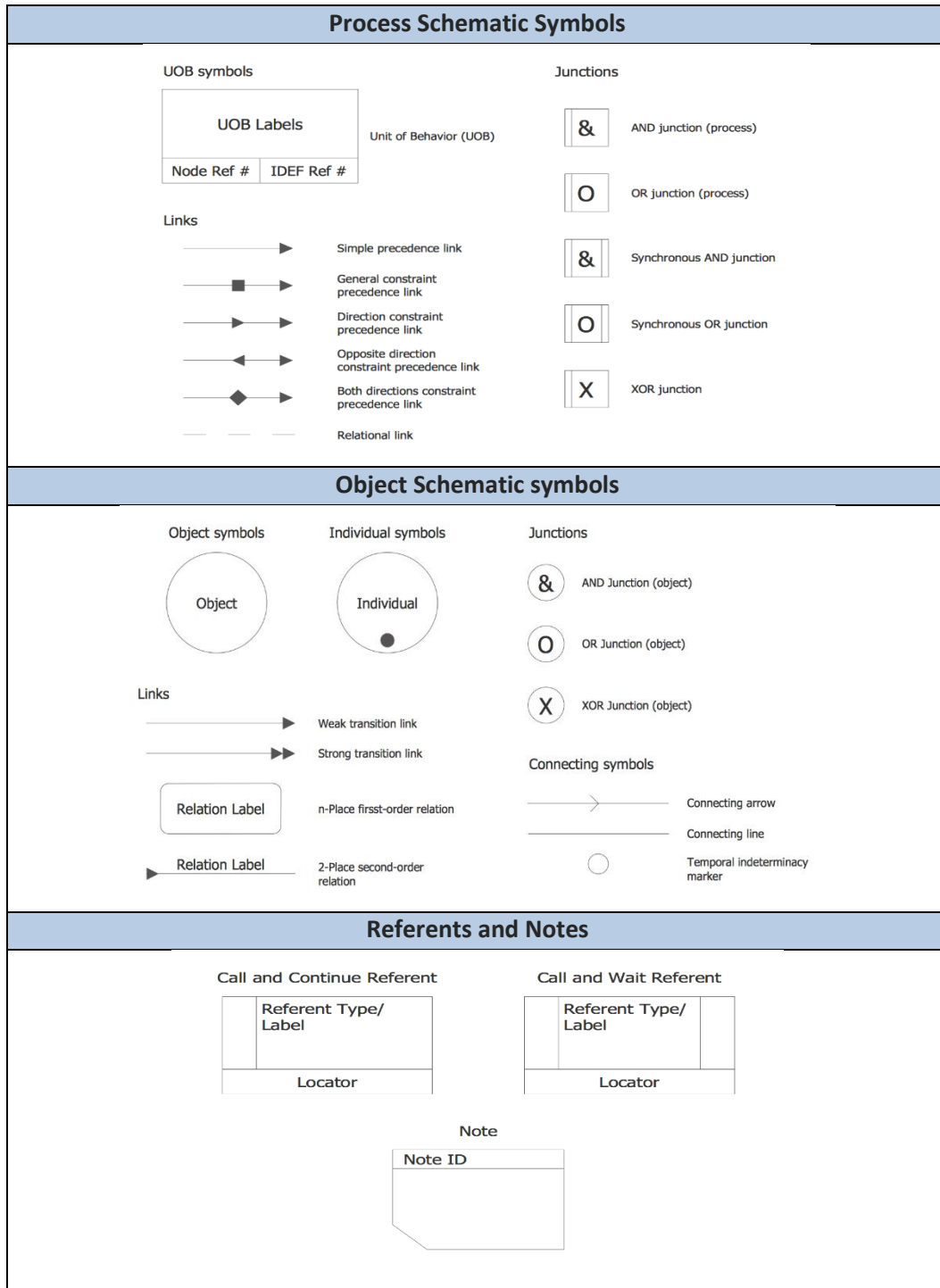


Figure 2.6 Symbols used for IDEF3 Process Descriptions by en.wikipedia.org
(UOB stands for Unit of Behaviour)

An example of IDEF3 description of a process using the process flow description and the object state transition description is shown in figures 2.7 and 2.8 respectively.

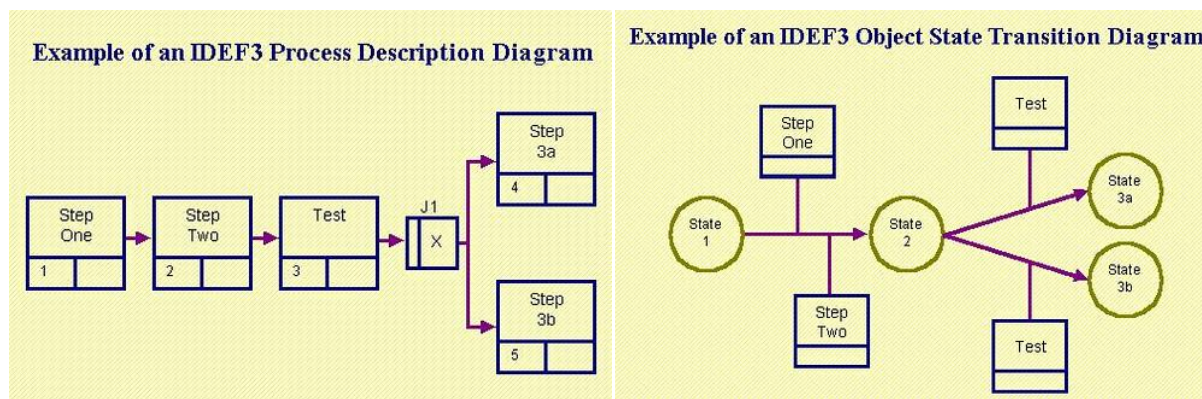


Figure 2.7

*Example of process flow description by
it.toolbox.com*

Figure 2.8

*Example of object state transition
network description by it.toolbox.com*

Of course, every UOB can be decomposed as it can have associated with it both “descriptions in terms of other UOBs” and a “description in terms of a set of participating objects and their relations”. The former is referred as a decomposition of a UOB and the latter as an elaboration of a UOB. A decomposition is a diagram and may be a decomposition of some top-level UOBs in the scenario or it may be the decomposition of a UOB in a decomposition. Also, multiple views (decompositions) are allowed in IDEF3 for the same UOB. An elaboration is an element of the IDEF3 description that captures the objects that participate in a particular activity and the facts and constraints that are defined on these objects and on instances of that activity. Each element of an IDEF3 description can have an elaboration. It is in the elaboration that resource requirements of systems will be captured. IDEF3 is used in several areas such as Business Process Engineering and Reengineering, software process definition and improvement and even in the software development and maintenance.

2.2.1.3 Unified Modelling Language (UML)

Unified Modelling Language (UML) is described as a general-purpose, developmental, modeling language in the field of software engineering, that is intended to provide a standard way to visualize the design of a system. UML was originally motivated by the desire to standardize the disparate notational systems and approaches to software design and has its roots in the object-oriented programming methods.

Large enterprise applications contain infinite lines of code, so they must be structured in a way that enables scalability, security, and robust execution under stressful conditions, and their structure-architecture must be defined clearly enough that maintenance programmers can find and fix a bug that shows up long after the original authors have moved on to other projects. Of

course, a well-designed architecture benefits any program, and not just the largest ones. Another benefit of structure is that enables code reuse which is the capability of structuring an application as a collection of self-contained modules or components. Eventually, enterprises build up a library of models of components, each one representing an implementation stored in a library of code modules. When another application needs the same functionality, the designer can quickly import its module from the library. At coding time, the developer can just as quickly import the code module into the application.

Modeling must be an essential part of all software projects. A model plays the analogous role in software development that blueprints and other plans (site maps, elevations, physical models) play in the building of a skyscraper. Using a model, those responsible for a software development project's success can assure themselves that business functionality is complete and correct, end-user needs are met, and program design supports requirements for scalability, robustness, security, extendibility, and other characteristics, before any changes to code be difficult and expensive to make.

The main benefit of UML is that is not assumed any specific methodology for analyzing and designing when UML is used to express the results. In addition, a UML model can be transferred from one tool into a repository, or into another tool for refinement or the next step in your chosen development process. UML can be used for business modelling and modelling of other non-software systems. Business process modelling with UML can be considered as an extension of the UML-based modelling discipline, related to system modelling using the same notation. The types of the diagrams supported by UML are divided into three categories which are presented in table 2.3 below.

Category	Types of diagrams
Structure	Class, object, component, composite structure, package, deployment
Behavior	Use-case (used by some methodologies during requirements gathering), activity, state machine
Interaction	Sequence, communication, timing, interaction overview

Table 2.3 Types of diagrams supported by UML by www.uml.org

Structure diagrams emphasize on the things that must be present in the system being modeled. Since structure diagrams represent the structure, they are used extensively in documenting the software architecture of software systems. For example, the component diagram describes how a software system is split up into components and shows the dependencies among these components.

Behavior diagrams emphasize on what must happen in the system being modeled. Since behavior diagrams illustrate the behavior of a system, they are used extensively to describe the functionality

of software systems. As an example, the activity diagram describes the business and operational step-by-step activities of the components in a system.

Interaction diagrams is a subset of behavior diagrams which emphasize on the flow of control and data among the things in the system being modeled. For example, the sequence diagram shows how objects communicate with each other regarding a sequence of messages.

The two mainstream diagrams in business process modelling are the use-case and the activity diagrams. The first one extends the software system use case concept to model the business system while the second one is focused on business processes. The basic syntax for the use-case and activity diagrams is presented in figures 2.9 and 2.10 respectively.

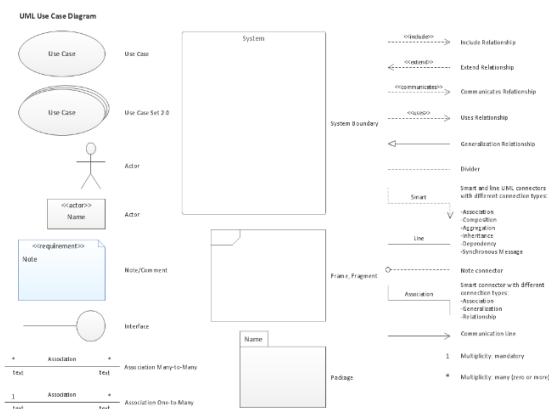


Figure 2.9

Basic syntax of use-case diagrams by www.conceptdraw.com

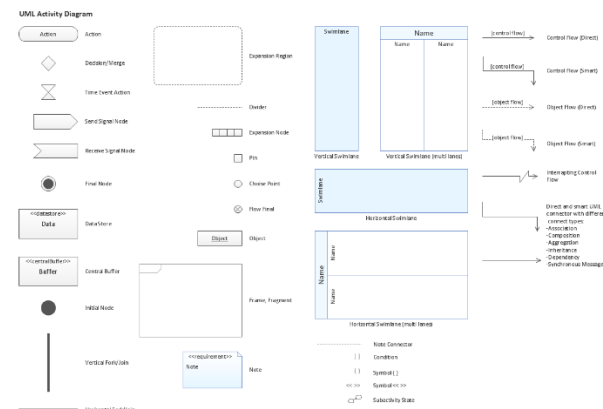


Figure 2.10

Basic syntax of activity diagrams by www.conceptdraw.com

The use-case diagram is used to define the behaviour of a system or other semantic entity without revealing the entity’s internal structure. Each use-case diagram specifies a sequence of actions, including variants, that the entity can perform, interacting with the actors of the entity. There is not a lot of notation as you see in figure 2.9, making them very easy to comprehend. Figure 2.11 shows an example of a use-case diagram modelling some processes of the university students’ system.

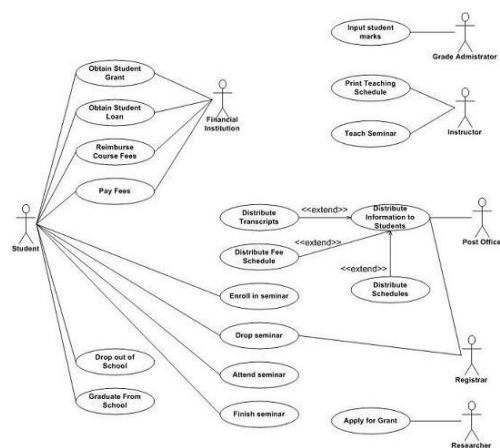


Figure 2.11

Use-case diagram modelling some processes of the university students’ system by www.agilemodeling.com

An activity diagram is the graphical representation of workflows of stepwise activities and actions with support for choice, iteration and concurrency. It is intended to model both computational and organizational processes along with the data flows intersecting with the related activities. In addition, it is typically used for business process modeling to visualize the logic captured by a single use-case or a usage scenario, or the detailed logic of a business rule. Consequently, it is considered as the object-oriented equivalent of flow charts. Figure 2.12 shows an example of an activity diagram coming from an email connection process.

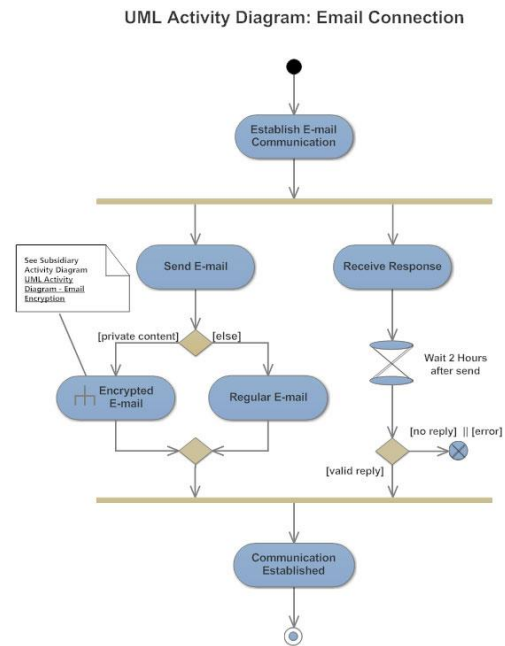


Figure 2.12

Activity diagram of an Email connection by www.smartdraw.com

2.2.1.4 Business Process Model and Notation (BPMN)

Business Process Model and Notation (BPMN) is a standard for business process modeling that provides a graphical notation for specifying business processes in a Business Process Diagram (BPD), based on a flowcharting technique very similar to activity diagrams from Unified Modeling Language (UML). The difference between BPMN and UML is that UML is object-oriented where BPMN takes a process-oriented approach which is more suitable within a business process domain. BPDs are commonly used to represent, analyze and implement the current (AS-IS) and improved (TO-BE) processes. The objective of BPMN is to support business process management, for both technical users and business users, by providing a notation that is intuitive to business users, yet able to represent complex process semantics. Its purpose is to model ways to improve efficiency, account for new circumstances or gain competitive advantage. The BPMN specification also provides a mapping between the graphics of the notation and the underlying constructs of execution languages, particularly Business Process Execution Language (BPEL).

Using the same visual language throughout the entire life cycle of the project development provides a great advantage for all project stakeholders making the development process more efficient. By having the business analysts and system developers using the same modeling concepts, the risk of costly errors related to different understanding of methodology concepts is significantly mitigated. BPMN has been undergoing a standardization push in the past few years. Such a standard provides businesses with the capability of understanding their internal business procedures in a graphical notation and gives organizations the ability to communicate these procedures in a standard manner. Furthermore, the graphical notation facilitates the

understanding of the performance collaborations and business transactions between the organizations. This ensures that businesses understand themselves and participants in their business and enable organizations to adjust to new internal and B2B business circumstances quickly.

BPMN supports modeling concepts only applicable to business processes. Other types of modeling for non-process purposes such as organizational structures, functional breakdowns or data models, are out of scope for BPMN. In addition, BPMN is not a data flow diagram, even though shows the flow of data (messages), and the association of data artifacts to activities.

BPMN defines a set of graphical objects, and rules indicating the available connections between these objects. The four basic element categories of BPMN are presented in table 2.4 and a summary of these elements is presented in figure 2.13.

Flow objects	Events, activities, gateways
Connecting objects	Sequence flows, message flows, associations
Swim lanes	Pools, lanes (sub-partitions of a pool)
Artifacts	Data objects, groups, annotations

Table 2.4 Basic element categories of BPMN by en.wikipedia.org

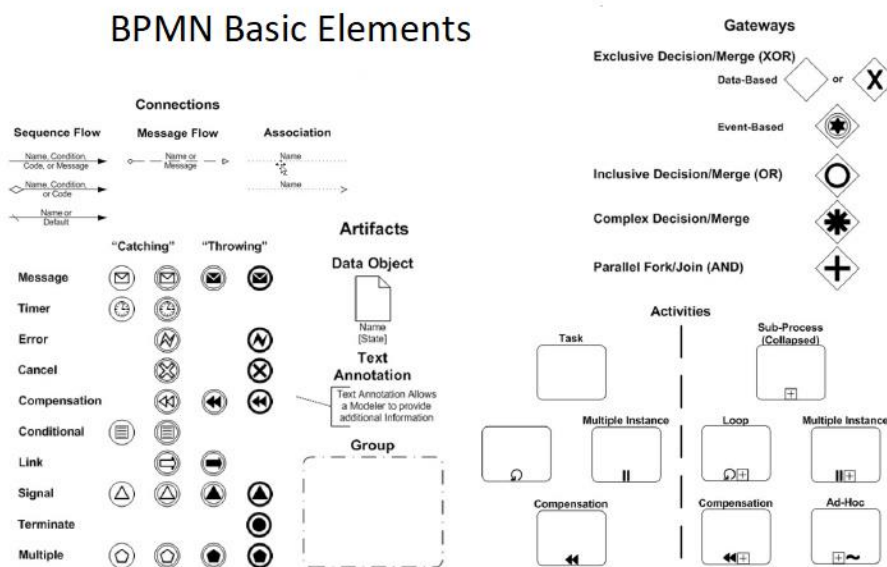


Figure 2.13 BPMN Basic Elements by Alexander Samarin

A BPD of a business process visually depicts a detailed sequence of business activities and information flows needed to complete this process. It consists of the start events, the processes to be performed within the process to be modelled and the outcomes of the process. Decisions and branching of flows are modelled by gateways. A gateway is like a decision symbol in the flowchart technique. A process can also contain sub-processes which can be modelled in another BPD

connected via a hyperlink to a process. If a process cannot be decomposed, it is considered a task, the lowest-level process. A “+” mark in a process denotes its capability for decomposing. An example of a BPD of an on-line auction system is shown in figure 2.14 below.

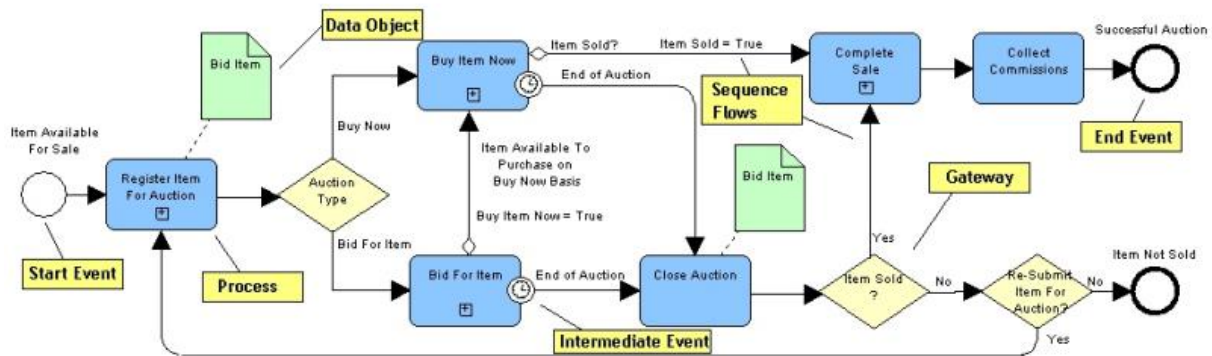


Figure 2.14 BPD of an on-line auction system by www.omg.org

Finally, you can drive further into business analysis by specifying ‘who does what’ by placing the events and processes into shaded areas called pools that denote who is performing a process. You can further partition a pool into lanes. A pool typically represents an organization and a lane typically represents a department within that organization (although you can make them represent other things such as functions, applications, and systems). An example of BPD containing pools is presented in figure 2.15 below.

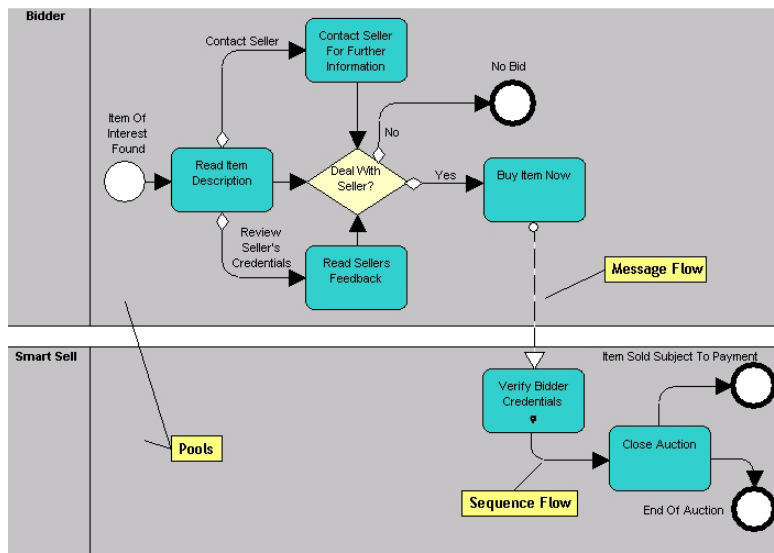


Figure 2.15 BPD example with pools by www.omg.org

Summing up, BPMN provides a standard, common language for all stakeholders, whether technical or non-technical: business analysts, process participants, managers and technical developers, as well as external teams and consultants. It has been developed to make the overall business lifecycle development process more efficient and ideally, bridges the gap between process intention and implementation by providing sufficient detail and clarity into the sequence of

business activities. Finally, since 2014, BPMN has been complemented by a new standard for building decision models, the Decision Model and Notation standard.

2.2.2 Mathematical/Formal Models

The main drawback of the business process modelling techniques described in the previous section is that none of them provides quantitative information to be used for analysis purposes. Zakarian (2001) points out that the process modelling techniques will be more attractive if formal techniques for analysis of process models are also provided. Formal models can define the process concepts rigorously and precisely so that mathematics can be used to analyse, extract knowledge from and reason about them. Koubarakis and Plexoudakis (2002) highlight the capability of formal models to be verified mathematically, as of high importance because this means that they can be proved as being self-consistent and have or lack certain properties. Van der Aalst *et al.* (2003) suggest that a formal foundation should be an integral part of business process models since it does not leave any room for ambiguity and the potential for analysis increases. Business process modelling lacks formal methods to support the business process model (BPM), according to Hofacker and Vetschera (2001). The reason is the qualitative nature of the business process elements and constraints; hence it is hard to parameterize them in a mathematical way, suitable for analytical methods, Tiwari (2001). There are few approaches that use mathematical models but there is not a common model to follow. Hence, every author found in literature, formulates the mathematical model of a business process according to the scope of his research. Furthermore, Hofacker and Vetschera (2001) in effort to provide analytical support for business process optimization (BPO), note that the description-oriented models such as the diagrammatic modelling techniques discussed in the previous section, assume that the sequence of the activities involved in a business process is taken for granted while formal techniques have the structure of a process model to be determined by the problem specification. This constraint according to the authors, is weaker than the precedence constraint usually considered in scheduling problems, since the same resources can be generated by different activities. Next, two business process modelling approaches are presented using a formal model. It is the author's opinion that these two are the most representative and comprehensive approaches found in literature so far, thus this research has been strongly motivated by them.

2.2.2.1 Modelling Approach by Hofacker and Vetschera (2001)

The first step towards analytical methods for business process modelling is owed to Hofacker and Vetschera (2001). They developed a general framework to represent administrative processes by setting mathematical constraints and a set of objective functions. These mathematical constraints define the feasibility boundaries of a business process. As objectives functions, they use an additive function to be minimized and the maximization of the minimum value found in all activities, but

any other objective function can also be used. The additive function simulates a cost function which sums the costs across the activities in a business process. Activities along with resources, are the main elements in a process model. Resources are divided into physical and information objects that flow through the system while activities demonstrate the transformation steps which use input resources and produce new ones as output resources. Each activity is represented by a node and uses one or more input resources and generates one or more output resources. Both input and output resources are represented by arcs connecting an activity to other ones. A business process has its own input and output resources called as global inputs and global outputs respectively. Additionally, a set of attributes is assigned to each activity for evaluation purposes for the entire process by aggregating the evaluations of the activities contained in the process, e.g. cost, duration or quality aspects. The sequence of activities is to be determined and the potential sequences are constrained by the requirement that resources must be produced by some activity before they can be used by other activities. Hence, there may be different sets of activities that when properly connected lead to different process models for the same process. The only difference is the sequence that these activities are executed with. For this reason, they consider a set of potential activities with different characteristics, e.g. inputs, outputs, attributes, which the process must be constructed from and try to find the subsets of these potential activities that comply with the problem specification.

The problem specification for the relationship between the activities and resources is based on the following three assumptions/constraints:

1. Each activity consumes exactly one unit of its input resources and generates one unit of its output resources
2. All input resources of an activity must be available before this can be executed
3. All output resources of an activity are generated when this is executed

For the first constraint mentioned above, Hofacker and Vetschera (2001) point out that this can be extended to allow for arbitrary input and output coefficients. It is up to the designer's perception and the examined process itself. In addition, they characterize the second assumption as non-critical because alternative input resources of an activity can be modelled in the same framework by defining additional potential activities and provide the following example:

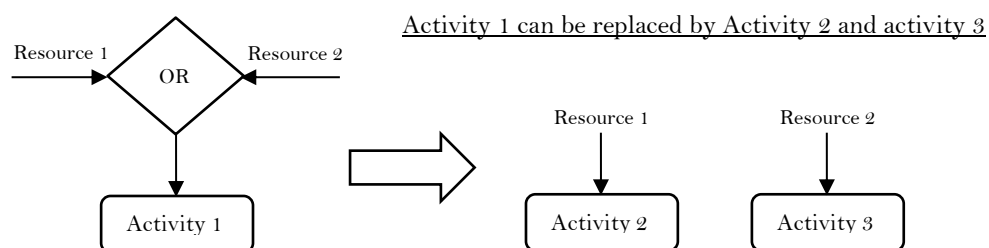


Figure 2.16 Example of a potential activity equivalence by Hofacker and Vetschera (2001)

The set of global inputs is available at the beginning of the process and the process must produce the global outputs. The assumptions taken by the authors according to the feasibility of a process design, are presented below.

1. For all activities contained in the process design, all their input resources are either contained in global inputs or are generated by other preceding activities. For physical activities, no other activity must consume the same unit of the resource.
2. All global outputs are generated by some activity contained in the process design and again, physical resources must not be consumed by other activities.

An example of a feasible business process design is shown below in figure 2.17.

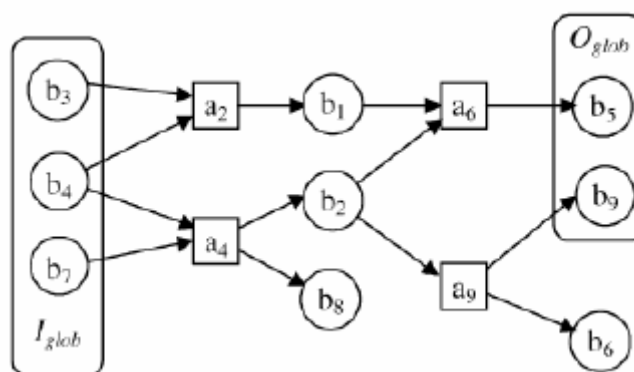


Figure 2.17 A feasible business process design by Hofacker and Vetschera (2001)

This attempt established the connection between a process design and a mathematical model and enabled the business process optimization (BPO). However, it is too generic to capture the aspects of the real-world processes. They directed attention to the real-world activities, resources and objectives. The outputs of real-world activities highly depend on their execution, e.g. a manufacturing activity can produce good or bad parts. This XOR junction must be incorporated somehow in the mathematical model. For real-world resources, there is no assumption that a resource is generated but not used in the subsequent process because of its cost for the company. E.g. a bad part in manufacture will not be thrown away but will take the way of rebuilding. By dividing resources into disposable and non-disposable ones, the authors managed to overcome this issue. If there are non-disposable resources not consumed in a process model, the whole model is infeasible. Finally, they recognize that real-world objectives may depend on the joint presence of activities in a model, hence you cannot simply evaluate them individually. Their proposal for this issue, has to do with assigning a presence indicator value to each activity for synergy identification purposes. Then, the corresponding coefficients to those synergies can be considered in the process evaluation.

2.2.2.2 Modelling Approach by Vergidis (2008)

An innovative formal specification and representation technique was proposed by Vergidis (2008) to enable the application of state-of-the-art evolutionary multi-objective optimization algorithms to business process optimization (BPO). This technique aimed to support a visual diagrammatic representation of processes and have a formal/mathematical underpinning so that quantitative measures can be extracted. The new in his research, was the development of the Process Composition Algorithm (PCA) to compose algorithmically business processes based on specific requirements and fill the gap between the visual and the quantitative perspective of business processes. PCA also plays a major role in business process evaluation and it will be extensively described later in this thesis.

This approach focused on business processes found in the service industry, hence a business process itself is considered as a service and its outcomes are non-material equivalents of goods based on the service definition. The proposed specification included all the value-adding business processes operations performed within an organization. It captured business processes regarding to the functionalities that are involved instead of the steps that should be executed and emphasized accordingly on the flow and connectivity of the participating functionalities rather than on execution details. In addition, it allows the hierarchical structuring of business processes like diagrammatic modelling techniques do. This comes from the perspective of identifying the main functionalities within a business process since it implies that a strategic process and an operational process can be similarly perceived. Therefore, a functionality identified in a higher level can itself be a business process at a lower level.

The visual representation is made via a simple flowchart and the main elements of a process are the tasks, the resources, the attributes and the connectivity patterns. Tasks represent specific functionalities intended to perform core operations. The difference among tasks is found in the core operation they perform. Being properly connected and sequenced, perform the business operation of a higher-level business process. Resources are related to the input and output products of the tasks and the business process. They are transformed while flow through the tasks of a process to produce the process output resources. Vergidis (2008) has not assumed any specific nature or type for the resources. They also control the way tasks are connected within a business process and help in shaping the connectivity patterns occurring in the process design. Every task has also a few attributes which represent their measurable characteristics to be used for the evaluation of the business process design. During evaluation, the process attributes are calculated as the aggregation of the corresponding task attributes of the participating tasks in the design. Finally, the author considered connectivity patterns as essential for expressing recurring paths in a process and responsible for shaping the process design and involved the *Sequence*, the *Parallel*

execution (AND), the *Multi-choice (OR)* and the *Arbitrary Loops* in his approach. An example of the visual representation of a generic business process design regarding to this notation is shown in figure 2.18.

- Two rounded boxes marked as ‘START’ and ‘END’ appear in every design and denote the beginning and end of the process.
- The participating tasks are sketched as boxes.
- The resources are the connecting arrows that link the tasks.
- The patterns are depicted as follows:
 - Sequence is sketched as the connecting arrow between two tasks
 - Parallel flow (AND) is sketched as box
 - Multi-choice (OR) is sketched as rhombus
 - Arbitrary loops (GOTO) are sketched as arrows pointing backwards

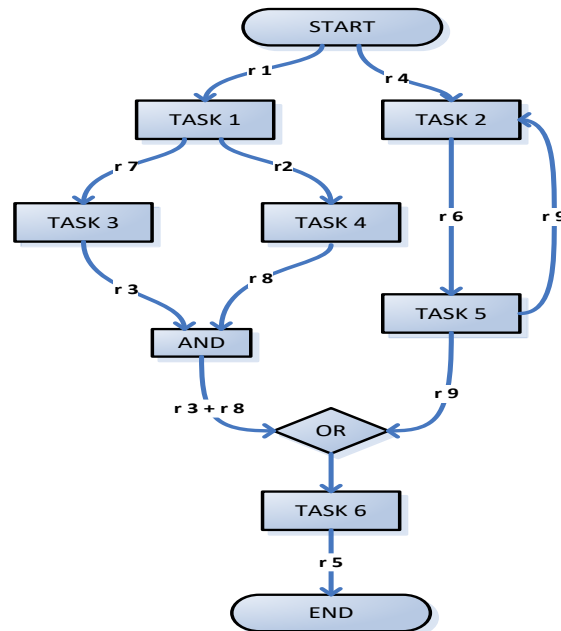


Figure 2.18 A generic business process design representation by Vergidis (2008)

The relationship between the activities and the resources assumes that if a resource is an input resource for a task, it cannot be an output resource for this task too and vice versa. Also, the feasibility of a business process is constrained by the following rules:

1. All process input resources are utilized by one or more tasks participating in the process design
2. All process output resources are produced by one or more tasks participating in the process design
3. Each task in the design is connected either with the process inputs, the process outputs or another task in the design

This approach is based on the same principles with Hofacker and Vetschera (2001) and extended their approach to the direction of the service industry. Vergidis (2008) examined business processes from the perspective of the functionalities involved in these processes and focused on the flow and the connections of these functionalities in the process design with respect to the process requirements. The process requirements are described by the above-mentioned assumptions and his main goal was to optimize business processes by searching and evaluating alternative feasible process models formulated by subsets of the library of potential tasks. He noticed that the three feasibility constraints yield a significant number of infeasible cases and identified the need of handling them algorithmically. Thus, he developed a *process composition algorithm*, called PCA, to tackle the infeasibility issues while trying to construct a feasible process

diagram. PCA will be extensively described later in another section. Infeasible designs emerge when:

1. One or more process input resources cannot be utilized from the tasks in the examined subset of potential tasks
2. One or more process outputs cannot be produced from the tasks in the examined subset of potential tasks
3. One or more tasks in in the examined subset of potential tasks cannot be attached to the process diagram based on its input and output resources

2.3 Business Process Optimization (BPO)

Business process optimization (BPO) is considered as the problem of constructing feasible business process designs with optimum attribute values such as duration and cost Georgoulakos *et al.* (2017). The business process modelling techniques described in the previous section, are strongly motivated by the need for business process improvement. According to Smith (2003), large organizations need to map their processes for two main reasons: One is to have a clear picture of the current situation and the flow of activities within the organization and second is to improve those processes efficiently to meet the organizational goals. Similarly, Grigori *et al.* (2004) acknowledge that organizations need to provide their processes with a high, consistent and predictable quality. They also identify as prerequisites for BPO that business processes should be correctly designed, their execution should be supported by a system that can meet the workload requirements and the process resources, e.g. human, material and non-material, should be able to perform their work items in a timely fashion. Therefore, an approach for BPO should clearly define and specify how optimization is perceived and which aspect of the process is going to be optimized. In this section, the author is going to present some of the optimization approaches found in literature and the modelling techniques that supported those approaches analytically.

2.3.1 Optimization Approach by Hofacker and Vetschera (2001)

Hofacker and Vetschera (2001) attempt to optimize the design of (mainly administrative) business processes. They introduce formal models for the business process design problem which can be used to analytically determine optimal designs with respect to various objective functions subject to several constraints. It is perceived to be the most comprehensive work towards BPO because three different optimization techniques have been examined along with the process formal model: mathematical programming, a branch and bound method and genetic algorithms.

2.3.1.1 Mathematical Programming Formulation

Their first attempt consists of the formulation of the process design to a mathematical problem. They use an additive function and several constraints to describe the problem and cover all its aspects. The objective function is minimized or maximized according to the optimization goal and the constraints describe and ensure the feasibility of the process in a mathematical formal way. As mentioned in the previous section, the main elements used in the process design are the activities and the resources. The mathematical constraints can be grouped into two major categories:

1. constraints related to input and output resources of each activity and
2. constraints regarding the time sequence of resources and activities.

Every process has a set of process input resources available and must produce the set of process output resources. The participating activities must be sequenced in such way that they use some resources as inputs and then produce resources that can be used as inputs by other activities until the set of process output resources is generated. The constraints of the first group ensure that input resources are available by activities to use and the set of process output resources is eventually produced.

Constraints belonging to the second group check the time sequence of activities and resources. Each activity has a starting point of time p and an execution duration. The input resources of an activity must be available before p and the output resources must be produced after $p + d$ time. The time when a resource becomes available is critical to the feasibility of the process design.

In order to set formally the constraints, the two authors introduce several variables and arrays that bind together the activities and the resources. That increases the complexity of the process model but also ensures its strict mathematical formality. In addition, it makes the model more flexible as a constraint can be eliminated to simplify a particular aspect of the model or extra constraints can be added to shape further the model. According to the experiments performed, the mathematical approach produced satisfying results but poor execution times.

2.3.1.2 Genetic Algorithms

Genetic algorithms have been successfully applied to complex problems in a variety of areas. Their advantage is that they maintain a population of possible solutions to reach feasibility and this makes them powerful. Another significant advantage is their extendibility to optimize a problem under more than one criterion. Multi-objectivity makes genetic algorithms a flexible methodology that can be applied to any optimization problem.

A genetic algorithm imitates the process of natural evolution to find an optimal solution. It works on many solutions in parallel, where each solution corresponds to an individual in the population. Each solution is represented by an appropriately coded string, its *genome*. A *mutation* operation changes the values of randomly chosen positions of that string. The resulting mutated individuals are then selected for mating. A *crossover* operation exchanges information between two individuals. Finally, the *selection* operation selects randomly the superior solutions to form the new generation. The selection probability depends on the objective function value, and the process continues until some pre-defined termination criteria are fulfilled.

The business process design described before, must be solved with respect to several constraints. The authors chose between two approaches to deal with the constraints within a GA framework. In the first approach, a penalty term for constraint violation is added to the original objective function. The second approach modifies the genetic operators to limit the search space to feasible solutions. This approach is appropriate if feasible changes can be easily determined. They decide to follow the first approach as the second would require extensive computational effort.

An individual, i.e. a (not necessarily feasible) process design P , is represented using a three-dimensional cube C . The cube C can be regarded as an extension to the adjacency matrix of a (meta-) graph. An element $C_{i,j,r}$ of C represents an arc between two distinct activities a_i and a_j , which are related by resource b_r . Figure 2.19 illustrates the representation of a process design.

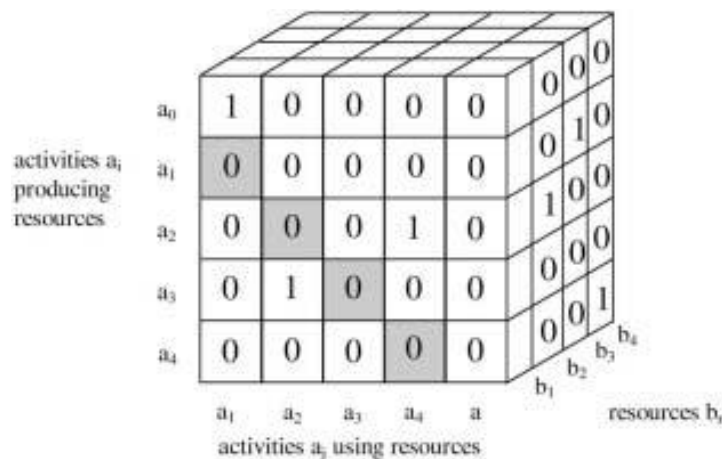


Figure 2.19 Representation of a design alternative by Hofacker and Vetschera (2001)

The information conveyed in the cube of figure 2.19 can be interpreted as follows: Resource b_1 is produced by activity a_3 and consumed by activity a_2 . Activities a_0 and a_∞ are two artificial activities added by the authors in every cube to represent the activity producing the global inputs and the activity consuming the global outputs respectively.

During mutation phase, a modification is implemented considering that only those elements of C can be assigned a positive value, representing a possible flow of resources between two activities. A supplementary cube C^P contains the maximum mutation probability for the respective elements of the solutions. Next, during crossover phase, pairs of cubes are selected from the population and they are cut by the same randomly determined plane. The two disjoint parts of the cubes are exchanged, and the new individuals are assigned to the new population. Finally, the fitness evaluation of each solution is made and then the selection of the next-generation individuals is made according to their fitness values. The fitness values of an individual are determined by the value of the original function, the penalty term that accounts for a possible infeasibility and a term accounts for the number of cycles formed in the design alternative. Cycles may be generated because of the selected mutation operator. The equation of fitness values is as follows:

$$f(P) = \sum_{i: a_j \in P} v_i + \lambda_1 \sum_{i: a_j \in P} \sum_{b_r \in I_i} miss(a_i, b_r) + \lambda_2(cyc)$$

Miss function returns the value 1 when the resource b_r which is an input resource for the activity a_i is not available. The parameters λ_i are used to calibrate the importance of penalty terms. The authors also point out that the calculation of the number of cycles in a design alternative P significantly increases the computational effort of evaluating the fitness function.

The initial tests have shown weak performance for genetic algorithms. The main issue was that the genetic algorithms could not maintain the feasibility of a design alternative in a tightly constrained problem as the *business process optimization problem*. The design of a process requires activities to be ordered so that all inputs of an activity are generated by preceding activities. The feasibility cannot be maintained by the operations of the genetic algorithms and therefore it is incorporated via the penalty terms in the fitness function. For this reason, the authors suggest that in a highly constrained problem an algorithm which maintains feasibility must search a much smaller space, leading to better performance.

2.3.2 Optimization Approach by Vergidis (2008)

Vergidis (2008) proposed a *business process optimization framework* (BPO_F) to capture, visualize and express a business process design in a quantitative way that allows Evolutionary Multi-Objective Optimization Algorithms (EMOAs) to generate a series of alternative optimized designs. He uses two additive functions, feasibility constraints, two matrices (TAM & TRM) and a process composition algorithm (PCA) to describe the problem and produce feasible and optimized design alternatives. The objectives can be maximized or minimized based on the optimization plan. The feasibility constraints ensure the validity of the process and as mentioned in the previous section,

the main elements used in the business process design are the tasks, the resources and the attributes. One of the used matrices is the *Task Attributes Matrix* (TAM) and captures the *attribute values of the tasks* in the design to facilitate the calculation of the process fitness values, hence the evaluation of the design. The second matrix is called *Task Resources Matrix* (TRM) and captures the *sequence of the tasks* and the *connectivity patterns* formulated in the process design by mapping their input and output resources. The relationship between the tasks in the design and the resources is denoted by the value of the corresponding cells (task, resource).

- If TRM_{ij} equal to 1, the resource belongs to the set of input resources of the task
- If TRM_{ij} equal to 2, the resource belongs to the set of output resources of the task
- If TRM_{ij} equal to 0, the resource belongs neither to the set of input resources nor to the set of output resources of the task

Attributes \ Tasks	A ₁	A ₂
	Task 1	100
Task 2	120	302
Task 3	117	324
Task 4	178	308
Task 5	145	356
Task 6	157	389
PROCESS	817	1979

Figure 2.20 Example of TRM by Vergidis (2008)

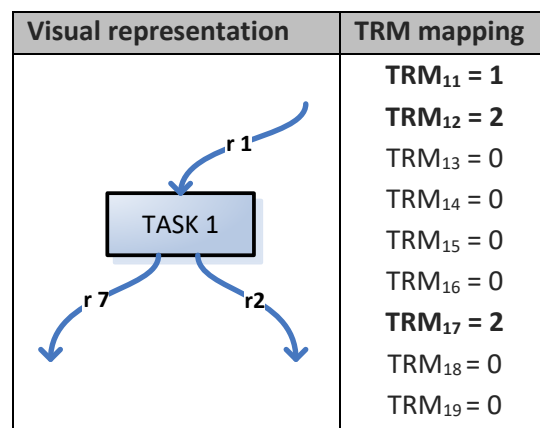


Figure 2.21 TRM mapping for task 1 by Vergidis (2008)

It is obvious that TRM provides the basis for reproducing the business process design based on the process requirements. However, the transformation of the quantitative representation of a business process into a visual diagram, is not trivial because the quantitative perspective cannot ensure the feasibility of the business process design based only on the TRM and the process input and output resources. A process design can only be transformed into a valid diagram, if the process is feasible. Figure 2.22 provides the basic pseudo-code to construct the visual perspective based on the quantitative perspective.

```

1. START with the process input resources
2. Attach the tasks that accept the resources as inputs based on the TRM
3. Draw the output resources of the attached tasks based on the TRM
4. IF the process output resources are produced,
   THEN the process design is complete (END)
   ELSE GOTO to step 2
    
```

Figure 2.22 Pseudo-code for constructing the visual representation perspective by Vergidis 2008

The pseudo-code in figure 2.22 assumes that the process design mapped in TRM, is feasible. However, this is rarely the case as the tasks in TRM might have been selected in an arbitrary way. This led the author to identify the need for an algorithm that can compose the visual perspective of a business process design, ensuring its feasibility based on the quantitative perspective. Hence, he developed an algorithm to construct the business process diagram, given its quantitative representation, and check whether the result corresponds to a feasible business process or not. This algorithm is called *Process Composition Algorithm (PCA)* and will be presented below.

2.3.2.1 Process Composition Algorithm (PCA)

The *Process Composition Algorithm (PCA)* provides the bridge between the visual and the quantitative perspective. It composes the visual diagram of a process, stored in TRM, in a way that the design captured by both representation perspectives is feasible. In addition, if it is provided with multiple TRMs – solutions by the EMOAs, it can generate alternative designs as it examines each solution individually.

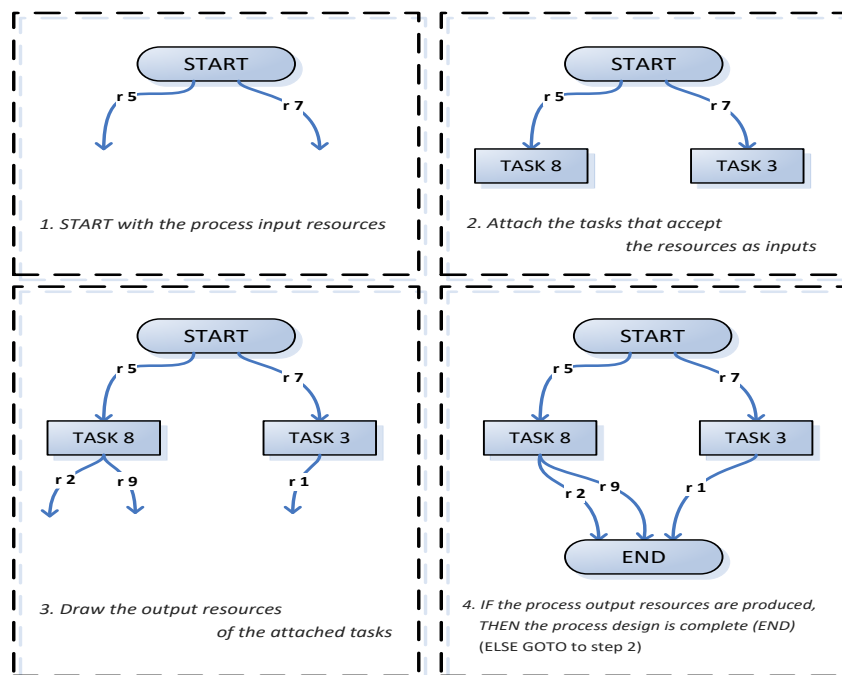


Figure 2.23 Business process graph elaboration by Vergidis 2008

Figure 2.23 shows how a business process design can be elaborated based on the pseudo-code in figure 2.22. A feasible business process is one that starts with the process input resources and by properly connecting the tasks in the TRM produces the requested process output resources.

The feasibility constraints presented above, yield a significant number of infeasible cases:

- One or more process input resources cannot be utilized from the tasks in the TRM
- One or more process output resources cannot be produced from the tasks in the TRM

- There is no task in TRM than can be attached to the process diagram based on its input and output resources

These cases of infeasibility result in a high probability of infeasible solutions found by PCA and this happens even for a large size of the task library. The first two cases where one process input cannot be utilized, or one process output cannot be produced, are inevitable. The third case of infeasibility may happen even in the case that all process inputs are utilized, and all process outputs are produced and is the most frequent. PCA attempts to tackle the infeasibility issues and construct a feasible process diagram.

1. Process input and output requirements (R_{in} and R_{out})
2. Participating tasks in the design (TRM)
3. Task library (N)

Figure 2.24 PCA requirements by Vergidis 2008

Figure 2.24 shows the requirements of PCA. The process requirements in the form of the process input and output resources are required as the termination criteria. The algorithm attaches tasks to the process design until the process inputs are utilized and the process outputs are produced as described in figure 2.22 too. The second requirement is TRM that contains the tasks that form the design for an individual solution. PCA will add tasks to the design from TRM and check whether they form a feasible solution. Finally, the task library is essential to modify or repair the design. Because of the high probability of infeasibility during the composition of a process design, PCA uses the task library to repair the design in order to make it feasible or to improve a feasible process design by replacing tasks with better attribute values.

The outputs of PCA are shown in figure 2.25. The main output of the algorithm is the business process design which is composed and represented as a directed graph. The nodes of the graph are the participating tasks and the edges represent the connecting resource. The graph is directed to reflect the flow of the resources between the tasks. The second outcome of PCA is the updated set of tasks that participate in the process design based on the execution of the algorithm.

1. Business process design (process graph)
2. Updated set of tasks in the design (N_d)
3. Degree of Infeasibility (DOI)

Figure 2.25 PCA outputs by Vergidis 2008

TRM represents the individual solution and PCA translates it into a process design. During the execution of PCA, TRM is updated for the following reasons:

1. The *elimination* of tasks in TRM that cannot be attached to the process diagram during its composition and
2. The replacement of tasks in TRM with tasks in the library that make the composed design feasible.

Last but not least, *Degree of Infeasibility* (DoI) is the third output of PCA. DoI was introduced by Vergidis (2008) as a metrics for process design infeasibility. Measuring the infeasibility of a design is of high importance in order for the different process designs to be compared and evaluated. BPO_F works with EMOAs which operate with a population of solutions. These solutions are evaluated based on their process attribute values. At any generation during the optimization process, the probability of a solution to be infeasible is very high and DoI is used for selecting the 'less' infeasible solutions and preserving them in the population as they have a better chance of evolving towards feasible solutions during the optimization process. DoI is based on three main factors and is calculated as:

$$DoI = n_{in} + 5r_{out} + 3r_{in}$$

For each infeasibility case, DoI assigns a different weight that reflects its relative importance and frequency. For every task inserted from the library of tasks in the process design, DoI is increased by 1 (n_{in} = total number of tasks inserted from the library). This infeasibility case is considered a frequently occurring one during the design composition, hence its weight. For every process output resource not produced, DoI is increased by 5 (r_{out} = total number of process output resources not produced). Vergidis (2008) considers this case as the most important one for the feasibility of a process design. The production of all process outputs serves as the termination criterion of PCA for a feasible process design. Finally, for every process input resource not utilized, DoI is increased by 3 (r_{in} = total number of process input resources not utilized). This case is as important as the previous one although the weight here is less than the output resources. Vergidis (2008) deems that the production of all output resources means that at some point all process input resources are utilized. For one or more input resources to be missing it means that the corresponding tasks were omitted during the last stage of PCA and thus the penalty is less. As each individual solution – process design carries a DoI, the feasibility comparison among the designs generated by PCA, is straight-forward. A feasible process design has zero DoI.

In figure 2.26 Vergidis (2008) presented the mains steps of the *Process Composition Algorithm*. PCA constructs a process graph and traverses it to ensure that it meets the process requirements. In

the graph, each task is represented as a node and there are two artificial nodes, the 'START' node with the process input resources and the 'END' node with the process output resources. These nodes facilitate the connection of the process input and output resources with the participating tasks in order to produce a process design that meets the process requirements. The graph is elaborated with the breadth-first strategy using the concepts of 'parent' and 'child' levels. The 'parent' level consists of the nodes already inserted in the graph and the 'child' level is the one where the new tasks are added in the design based on the output resources of the tasks in the 'parent' level. Once the elaboration of all tasks in the 'child' level is completed, it becomes 'parent' level for the graph elaboration to proceed.

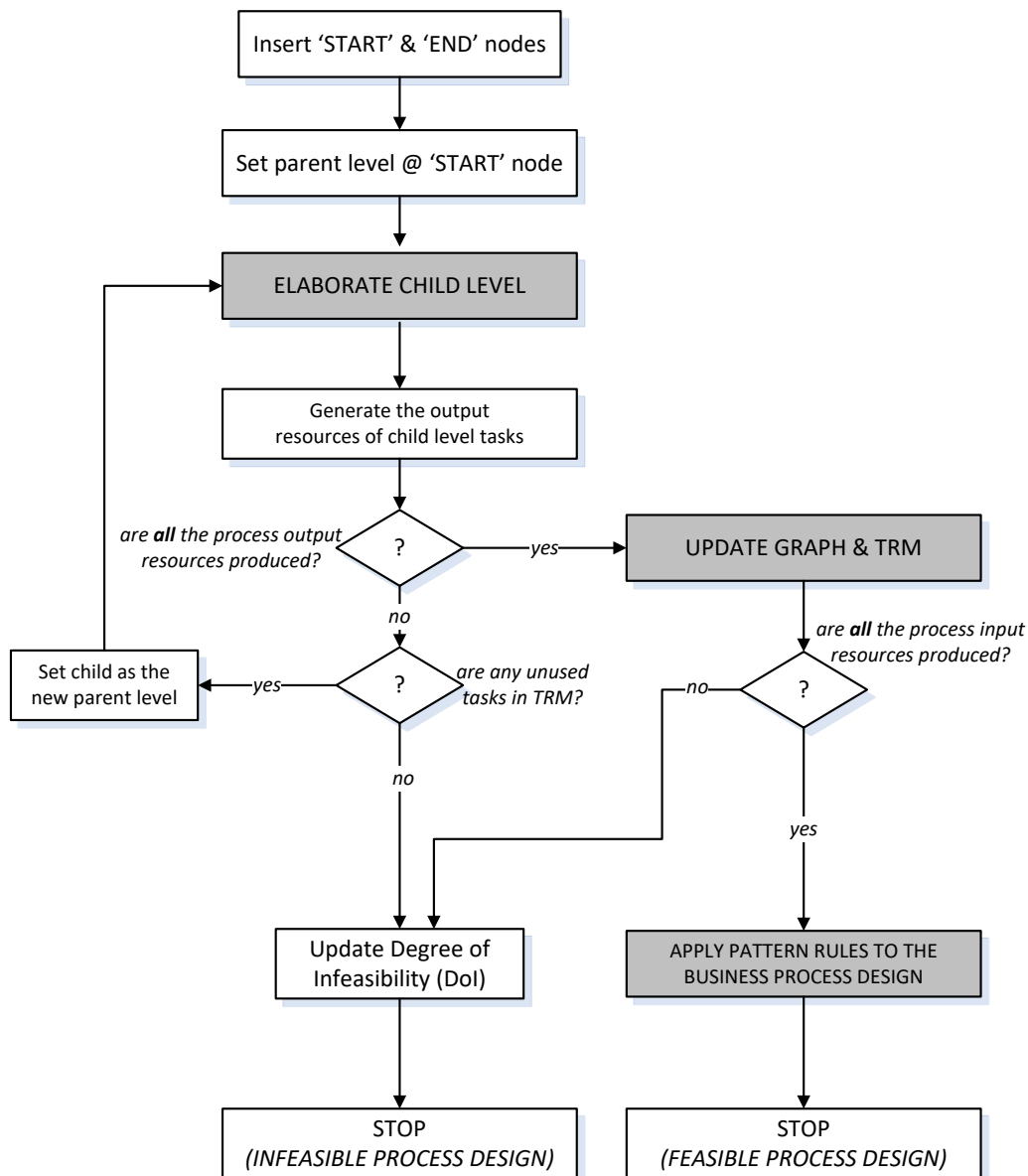


Figure 2.26 Main steps of PCA by Vergidis 2008

PCA starts by inserting the artificial nodes 'START' and 'END' to an empty graph. The 'START' node is initially marked as the 'parent level'. Then, the algorithm visits all the nodes in parent

level one by one in order to elaborate the child level. Once the child level elaboration is completed, the output resources of the recently attached tasks along with the unlinked output resources of previous tasks are checked to find out whether they contain the process output resources. In the case that not all the output resources are produced and there are unused tasks in TRM, the tasks in ‘child’ level become the new ‘parent’ level and the elaboration process is repeated. If there are no unused tasks in TRM then for every output resource that has not been produced there is a penalty attached to the design and DoI is updated accordingly.

In the case that –at some stage of the elaboration process– all the process output resources are produced, TRM and the graph are updated. The update process involves two parts: (i) the elimination from TRM of any tasks that have not been inserted in the process design, and (ii) the elimination of graph nodes (tasks) that do not contribute to the production of the process outputs. After the update, PCA checks whether all the process input resources are produced. Some of the tasks that were utilizing the process inputs might not have contributed to the process outputs and therefore are removed from the design. In the case that one or more process inputs are not utilized, there is a penalty attached to the design and DoI is updated accordingly. In the case that all the process inputs are produced, the design is marked as *feasible*.

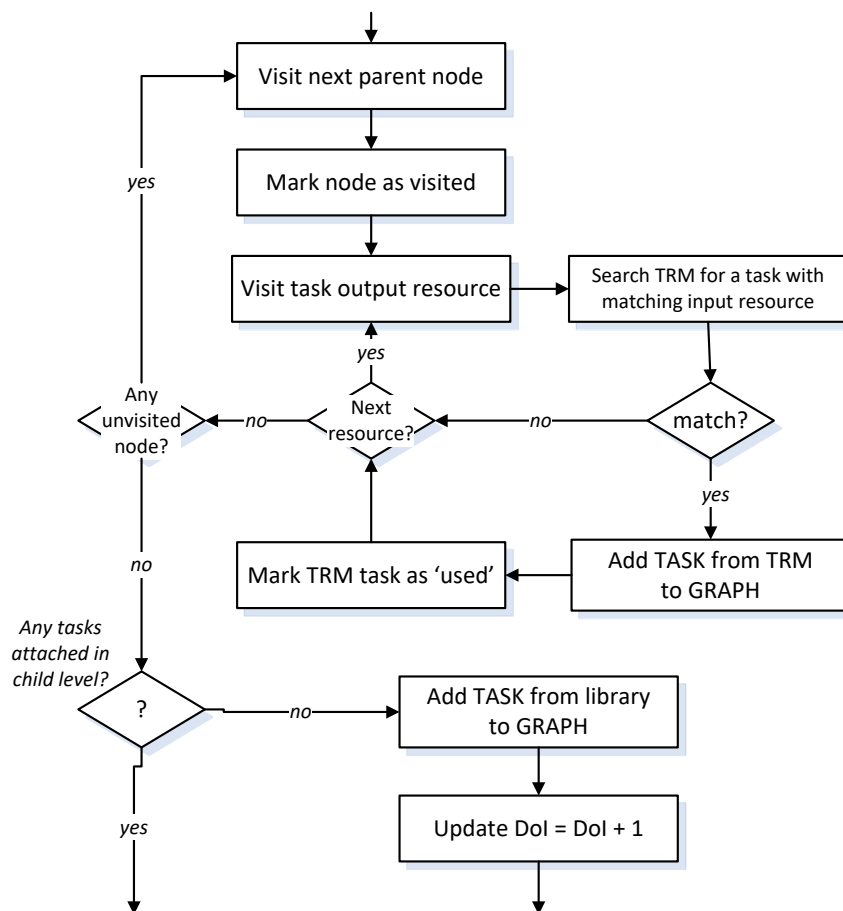


Figure 2.27 Algorithm for “ELABORATE CHILD LEVEL” phase by Vergidis (2008)

In the child level elaboration phase, tasks are inserted in the graph and attached to the output resources of the parent level nodes. For every node in the ‘parent’ level’, all the output resources are visited. For every output resource the algorithm checks in TRM to find a task with at least one matching input resource. If a task with common resource is found, it is inserted in the graph, linked with the parent task and added to the ‘child’ level set. There might be resources for which there is no matching task. In case that there is no matching task, the algorithm proceeds to the next output resource of the parent level. When the algorithm reaches the last output resource of the last ‘parent’ level task, it checks whether there are any tasks attached in the ‘child’ level. In the case that there are no tasks inserted in the ‘child’ level, the algorithm attaches a matching task from the task library in order to continue with the graph elaboration process. As a result, a penalty is attached to the design and DoI is updated. Every task that is added to the design is linked not only with the parent task but also with any task with which it has a matching resource.

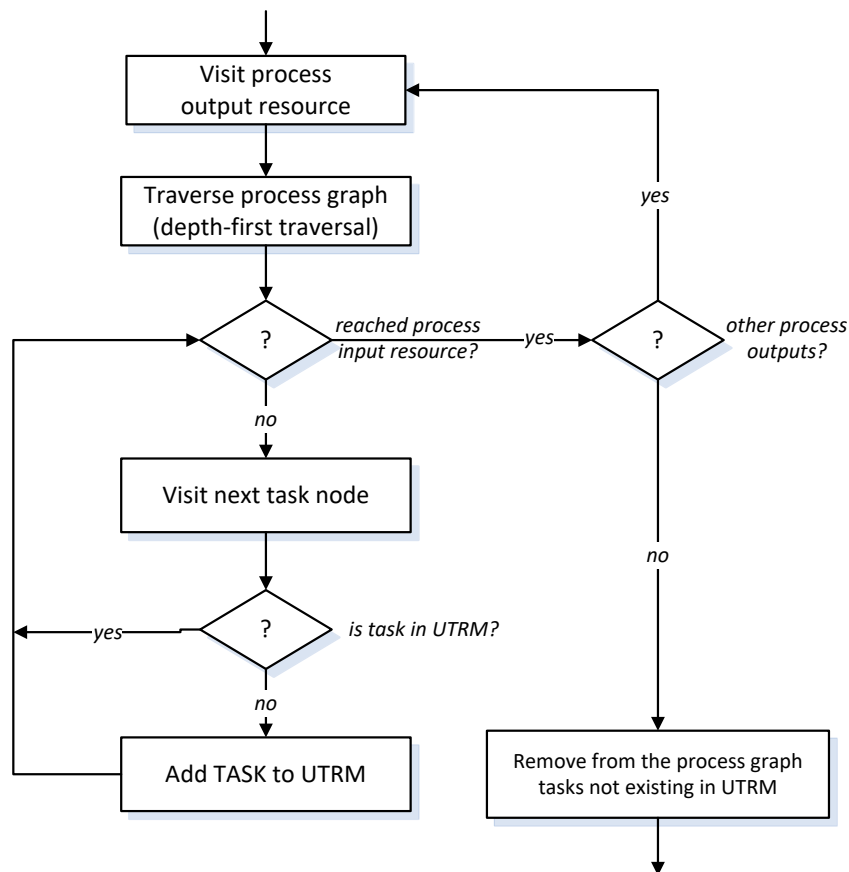


Figure 2.28 Basic steps for the “UPDATE GRAPH & TRM” operation of PCA by Vergidis (2008)

When all the process output resources are produced, PCA removes from the graph any nodes that haven’t contributed to the process outcome. The result of PCA is, in the best case, a feasible business process design in which all the tasks in the design are linked together utilizing the process input resources and producing the process output resources. PCA does not scrap the infeasible solutions but it repairs them by utilizing the task library or attaches a penalty to

demonstrate their DoI. In the evolutionary approach that he follows, infeasible solutions can lead to feasible ones as they evolve over the optimization generations.

2.4 Summary

This chapter examined the basic aspects regarding business process definition, modelling and optimization. As mentioned in chapter 1, this research attempts to provide a new evolutionary multi-objective optimization framework for business processes. This chapter provided an overview of the most common definitions for business processes, the main techniques for business process modelling and the most representative approaches for evolutionary multi-objective business process optimization. This literature survey enables the identification of the research aim and objectives in the next chapter.

CHAPTER 3

Aim & Objectives

This thesis aims at contributing to and extending previous approaches for *Evolutionary Multi-objective Business Process Optimization*. This chapter specifies and discusses the aim and objectives of this research. Based on these, the research deliverables and the expected gains are also elaborated and discussed.

3.1 Research Aim

This thesis is based on the studying of the most comprehensive approaches found in literature for *business process optimization*. The author aims at signifying the room for improvement and extension of these approaches and introduces a new version of an existing framework for BPO (eBPO_F), which includes a pre-processing stage and a new process composition algorithm. The pre-processing stage attempts to increase the ability of the new framework to find the optimized design alternatives by removing the tasks that would never be part of such a design. The new process composition algorithm which is called PCA-II, makes eBPO_F capable of fulfilling more real-life constraints during the design composition and handling more complex problems. The pre-processing stage and PCA-II are described in the next chapters.

3.2 Research Objectives

The aim of this thesis can be broken down to the main objectives that can be summarized as follows:

1. Studying and understanding previous approaches for BPO, namely by Hofacker and Vetschera (2001) and Vergidis (2008)
2. Reviewing the results of BPO_F reported by Vergidis (2008)
3. Improving, developing and extending BPO_F on the same principles using Python
4. Introducing a pre-processing stage for BPO problems
5. Proposing a novel design composition algorithm, PCA-II, for business processes
6. Validation and testing of the proposed framework

3.3 Research Methodology

The definition of the research methodology creates a starting point for each objective and shows the way for achieving them. In addition, by defining the research context, the expected outcomes can be further clarified.

The first objective is related with the good comprehension of the existing approaches that use EMOAs for BPO. Part of this research has already been carried out in the literature review chapter, where the most significant modelling and optimization techniques have been presented. In the following chapters some additional aspects of those approaches will be further discussed. It is the author's opinion that this objective is the most important because the whole research is based on the improvement and the extension of those approaches, especially the one owed to Vergidis (2008), and a high level of understanding will facilitate the author's intention.

The second objective is related with the review of BPO_F. As it has already been discussed, BPO_F is a complete framework for *business process optimization* problems, implemented by Vergidis (2008). This framework filled the gap between the visual and the quantitative representation of a business process and enabled EMOAs to generate a series of alternative optimized designs for BPO problems. It displayed satisfactory results with problems derived from the service industry and it will form the basis for the framework that will be developed in this research. The review of the results of this framework will facilitate the author to identify where there is room for improvement and the changes needed in order for the new framework to be able to handle problems from other domains apart from the service industry.

In this research, the author attempts to develop a new framework for BPO by improving and expanding the existing BPO_F framework which is implemented in Java. Java is a general-purpose, object-oriented, compiled programming language. It is an excellent option for developing a high-performance, robust and secure application whose compiled code will run on all platforms that support Java without the need for recompilation. However, it is not the best choice for numerical computing and quick prototyping. In addition, the need for compilation of Java makes debugging and code tracking much more difficult than in other programming languages. On the other hand, one of the most suitable programming languages for numerical computing and quick prototyping is Python. It is an interpreted, high-level, general-purpose, object-oriented programming language. The interpretation feature of Python facilitates debugging and easy code tracking and it is the author's opinion that this is an asset for numerical computing. Furthermore, python emphasizes on code readability, supports functional and imperative programming and along with Scipy, a Python-based ecosystem of open-source software for mathematics, science, and engineering, form an excellent choice for developing within the context of this thesis.

Consequently, the third objective of this research is to transfer the development and the maintenance of the existing framework from Java to Python based on the same principles.

The fourth objective is related to the efficiency of the EMOAs and the complexity of the BPO problems. The business process multi-objective optimization problems belong to the NP-hard problems, which indicates that both the efficiency of optimization algorithms and the quality of produced results rely upon the size of the examined problem. In general, business processes have many available alternatives for the participating tasks and these, in turn, can involve many different resources as either requirements or products of their utilization. The process, though, of algorithmically seeking the set of alternative non-dominated optimal business process designs can be particularly complex due to the numerous combinations that have to be produced and evaluated. A common practice that is followed by scientists to tackle such complexity issues, is the incorporation of a pre-processing stage in effort to reduce the amount of the data that must be processed during the solving process of a certain problem. This may involve utilizing unused information that comes from the problem itself or removing the faulty values of the data set according to the problem. Therefore, the author introduces a series of pre-processing steps, developed with the intention to be applicable to any business process dataset and aiming for increasing the efficiency of EMOAs.

The new in Vergidis' (2008) research was the introduction of the *process composition algorithm* (PCA). PCA was the missing part in the interpretation of a business process from the quantitative perspective to the visual perspective. Given the quantitative representation of a potential solution and the process requirements, PCA attempts to compose algorithmically a feasible business process design by implementing the necessary steps that create a diagram of a business process design using the proposed representation. It is also responsible for the evaluation of such a design and, as such, for the evolution of the population maintained by the EMOAs. The problems which Vergidis (2008) was involved with, came from the service industry and the top-down approach followed by PCA for composing the business process diagram, could handle such problems. However, one of the rules for defining a business process as feasible according to Vergidis (2008) is too loose. Specifically, it says that "*Each task in the design is connected either with the process inputs, the process outputs or another task in the design*". This rule may become stricter when dealing with problems from other domains apart from the service industry. For example, a feasibility constraint according to Hofacker and Vetschera (2001) is: "*All inputs must be available before an activity can be executed*". As it will be discussed later in this thesis, the logic of PCA is not consistent with that rule and may produce erroneous results. Therefore, another approach for business process composition must be followed and the fifth objective has to do with the development of a new algorithm (PCA-II) that will be able to compose more complex business process designs.

The final objective of this thesis refers to the testing of the revised optimization framework and the evaluation of the results. A set of real-life scenarios introduced by Vergidis (2008) is tested, but this time there are more constraints for a business process design to be feasible. In this way, the validity of the results can be achieved through the direct comparison of them against those produced by BPO_F.

3.4 Research Deliverables

After the directions of this research have been presented, the expected research deliverables can be listed below:

- ❑ A revised and improved version of the existing optimization framework, eBPO_F. This framework also enhances the reviewing of the produced optimized business process diagrams through the production of graphics that the reviewer will be able to interact with.
- ❑ A pre-processing methodology for BPO problems and the complete implementation of it, with the intention to be integrated in any BPO framework.
- ❑ A new process composition algorithm that gives the ability to the new framework to solve more real-life BPO problems.
- ❑ The results produced by the software tool that will be discussed and evaluated for their quality, performance and ability to provide realistic solutions in terms of alternative optimized business process designs against the results coming from the existing framework.

3.5 Expected Gains

The contribution of this research to the domain of *Business Process Optimization* can be divided into three directions. The first one has to do with the introduction of a new software tool for BPO problems. This tool constitutes the improvement of the existing one towards usability, I/O, interactivity and maintenance. Its code is separated in modules and the best practises have been considered throughout the development. The user is only responsible to provide the problem specification in a comma-separated file (.csv) and configure the environment that the tool will run in by setting some parameters. The users can choose the evolutionary algorithm, the population size, the operator probabilities, the number of generations etc. These parameters have been left at the user's will so that he can experiment with different configurations and choose the most suitable for the examined problem. At the end of the execution, the user will be able to see a graphic with the solutions found by the tool and interact with them to see the process diagram and compare different designs.

The second contribution to the domain of BPO is the proposal of a pre-processing algorithm for BPO problems. This algorithm aims to improve the efficiency of the evolutionary algorithms so as for them to produce better solutions. The improvement derives from clearing the problem dataset out of tasks that will never be part of the best solution or tasks whose features don't comply with the problem constraints. Another gain from such a pre-processing algorithm is that the tool can work efficiently without the need of a problem-specific dataset. This way, it can be used as a mining tool for business processes that could be formed by already existing tasks; it should only be fed with the dataset containing all existing tasks in the business.

The last contribution of this research is the new process composition algorithm for BPO problems. This algorithm gives the ability to the new framework to handle more complex problems since it follows a different approach for process composition than its predecessor by Vergidis (2008). The approach of PCA itself raised some inherent limitations that didn't allow the framework to produce feasible solutions for more complex problems. Such cases and limitations accompanying them will be further discussed in the following chapters. Finally, PCA-II has the notable asset to produce alternative feasible designs for a specific solution. The difference between such designs is only found in the utilization distribution of the resources across the involved tasks. Two alternative designs for a specific solution are presented in figure 3.1 below.

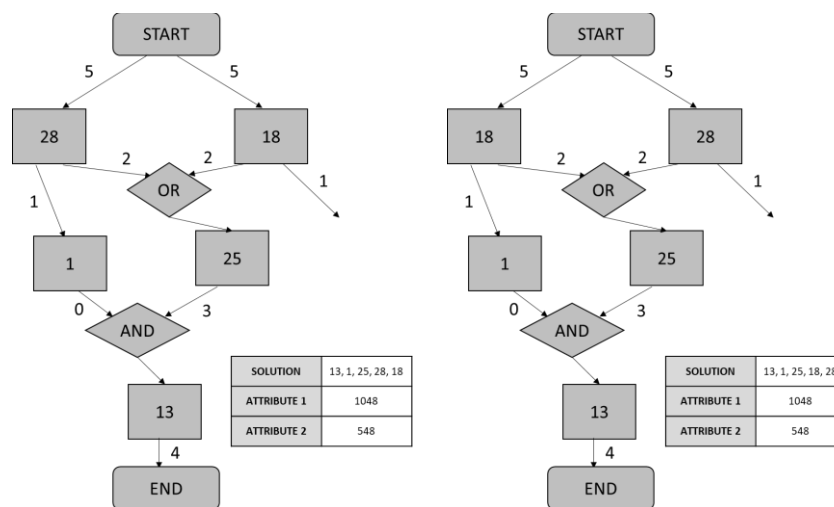


Figure 3.1 Alternative designs for a specific solution

Note that both the process designs above, are actually the merger of two different 4-task solutions, $[13, 1, 25, 28]$ and $[13, 1, 25, 18]$, and emerged from a new feature integrated in the new process composition algorithm. PCA-II can follow two different composition approaches for a solution examined, the non-greedy approach and the greedy one. The concepts discussed in this chapter will get clearer in the chapter where PCA-II is presented.

3.6 Summary

This chapter presented the aim of this thesis as the development of a new business process optimization framework based on previous approaches, capable of enhancing the efficiency of the employed EMOAs and suitable for complex problems. The aim is elaborated in specific objectives which detail the main actions of the research and the research methodology discusses the way for achieving them. Finally, the research deliverables are clarified to reveal the expected gains and the contribution of this research to BPO. The next chapter introduces the proposed pre-processing technique for business process optimization problems.

CHAPTER 4

Deploying Pre-Processing to eBPO_F

This chapter introduces a pre-processing technique for business process optimization problems. It starts by presenting both the purpose of the integration of such a technique as a mandatory part to the procedure of solving business process optimization problems, and the facts that were considered and led to the development of the proposed algorithm. Next sections present the steps of this algorithm and their pseudocode.

4.1 Purpose of data pre-processing

When someone deals with the solution of decision or optimization problems, encounters the difficulties that emerge from the computational complexity of them, such as the excessive execution time and the high memory usage. A common practice that is followed by scientists in those situations is the development of a pre-processing stage in effort to reduce the amount of the data that must be processed in the solution of a certain problem, utilizing unused information that comes from the problem itself or removing the faulty values of the dataset according to the problem.

4.1.1 Necessity of a pre-processing stage in BPO

The business process multi-objective optimization problems belong to the NP-HARD problems, so the efficiency of the optimization algorithms and the quality of the produced results highly depend on the size of the examined problem. Generally, business processes can have too many available alternatives for the participant tasks and these in turn can involve many different resources as requirements or as products of their utilization. Consequently, the process of seeking for the set of the alternative non-dominated optimal business process designs by an algorithm can be extremely difficult because of the vast amount of the combinations that must produce and evaluate.

In addition, the solution of NP-HARD problems, like business process optimization problems, is often approached by heuristic methods that try to find solutions which approximate the optimal ones, instead of exact algorithms. Heuristic methods may have a time limit to their execution or a limit to the evaluations that will take part during their execution. So, if these methods deal with

datasets, as much as possible, free of unnecessary and faulty data, they will produce better results. Specifically, for this thesis, some of the state-of-the-art genetic algorithms are used in effort to solve various multi-objective business process optimization problems. Most of the algorithms that belong to this category of genetic algorithms have limits within their execution, related with the number of evaluations or generations that will take part in their execution cycle or with the population size that they manage.

Furthermore, the proposed business process optimization framework (eBPO_F) can be used as a mining tool for business processes. Given a library of tasks, the process requirements and the related constraints, the framework is capable to produce the potential business process designs, if any, that conform to the problem specification. In this case, the pre-processing stage can be used as a fast way to see if the given library of tasks is suitable for producing at least one feasible business process design according to the problem specification without executing the main operation of the framework.

With this in mind, a series of pre-processes is proposed, combined in a pre-processing module aiming for cleaning any business process dataset of unnecessary and faulty data in effort to increase the efficiency of the used algorithms and examining if at least one business process design may be composed by the given dataset.

4.1.2 Extracting useful information from the business process problems themselves

Business process optimization problems always come with a set of rules and constraints that must be satisfied in order to get correct and accurate results. This set constitutes an integral part of the solution process itself and shapes the products of this process. Furthermore, if someone looks more carefully to this set, he will extract the necessary pre-processes that must be applied to any given business process dataset.

4.1.2.1 Tasks that their set of input resources is a superset of the set of the process output resources must be removed

Every business process has a set of process output resources that must be entirely produced at the end of its execution. So, the complete production of this set indicates the end of the business process execution and is one of the requirements that must be satisfied in order to have a feasible business process design. In case the set of the input resources of a task is a superset of the set of the process output resources, this means that this task is a potential task of a business process that follows the examined business process, thus it can't be part of a feasible optimized business process design of the examined problem; the input resources can't be utilized by a task before some other tasks produce them, and the complete set of the process output resources is produced at the end

of the business process. Therefore, if the library of potential tasks of a business process optimization problem also consists of such tasks, these tasks must be removed before the main operation starts executing. This action will prevent the evolutionary algorithm from unnecessary and ineffective executions related with potential solutions partially consisted of such faulty tasks.

4.1.2.2 Tasks that their set of output resources is a subset of the set of the process input resources must be removed

Every business process has also a set of process input resources that must be entirely utilized during its execution. This statement is another requirement that must be satisfied in order to have a feasible business process design. In case the set of the output resources of a task is a subset of the of the process input resources, this means that this task is a potential task of a business process that precedes the examined business process, thus it can't be part of a feasible optimized business process design of the examined problem. Hence, if the library of potential tasks of a business process optimization problem also consists of such tasks, these tasks must be removed before the main algorithm starts executing. This action will prevent the evolutionary algorithm from unnecessary and ineffective executions related with potential solutions partially consisted of such faulty tasks.

4.1.2.3 Tasks that require an input resource which is not produced by any other task or the set of process input resources must be removed

One of the rules related with the business process optimization problems is that the participant tasks in a feasible business process design must have all their input resources available before their execution. These resources can be obtained from the produced resources from the execution of some preceding tasks in the business process or directly from the set of the process input resources. Hence, if an input resource of a particular task doesn't belong to the set of the process input resources or it can't be obtained from another task in the library, this task must be removed from the library.

4.1.2.4 Tasks that none of their output resources is required as input by another other task or the set of the process output resources must be removed

Another rule related with the business process optimization problems is that a participant task in a feasible process design must give at least one of its output resources as an input resource to one of the succeeding tasks in the business process or directly to the set of the process output resources, otherwise it is needless for the purpose of the specified business process. So, the library of tasks must be searched for such tasks and if any of them is needless according to the input resources of the other tasks in the library, it must be removed.

4.1.2.5 Tasks that have the same input and output resources so that they can be classified, and are dominated by other tasks of the same classification according to their attribute values and the kind of the examined problem must be removed

The purpose of the proposed optimization framework is to find the optimal business process designs according to the attribute values of the participant tasks. Usually, in the library of potential tasks, some of them can be grouped in categories. The tasks of a certain category have the same input and output resources and the only difference among them is that they have different attribute values. The participation in potential solutions of a categorized task that has worse attribute values than the others in the same category according to the kind of the optimization we expect, leads to feasible suboptimal business process designs because if it is changed with another task that belongs to the same category, this action will lead to better solutions. Therefore, the final pre-process of the pre-processing module is to find whether there are tasks in the library that can be grouped in categories and if there are tasks in these categories that are dominated by others in the same category. If there are such inefficient tasks, these must be removed from the library of tasks in order to increase as much as possible, the efficiency of the used evolutionary algorithms and gain better solutions.

4.1.2.6 Examine whether the problem specification is valid for the given dataset or not

The process requirements in terms of the process input and output resources may not be suitable for the given library of tasks. This means that the process input resources are not required by any task in the library or/and at least one process output resource cannot be produced by any task in the library. Hence, the set of the process input resources is totally needless or/and the set of the process output resources cannot be entirely produced during a business process design composed by a subset of the tasks in the given library. As discussed above, both cases lead to an infeasible business process design. This could possibly happen for two main reasons. The first reason is the wrong configuration of the process requirements. This may be due to the user's mistake or the framework is used as a business process mining tool and a suitable set of process requirements for the given library of tasks hasn't been found yet. The second reason has to do with the pre-processes that take place in the pre-processing stage. After the execution of the pre-processes described above, some tasks may be removed from the library so that the remaining tasks cannot meet the process requirements. Therefore, an investigation must take place in the library of tasks at the end of the pre-processing stage whether the process input resources or/and the process output resources can be useful and be produced respectively or not, in order for the main optimization operation of the framework to be performed or not.

4.2 Main Steps of Pre-Processing

The pre-processing operation consists of five distinct sub-processes that perform the operations needed in the library of tasks as they were discussed above. In figure 4.1 below, the flowchart of the pre-processing algorithm is demonstrated as it has been designed by the author in the context of this thesis.

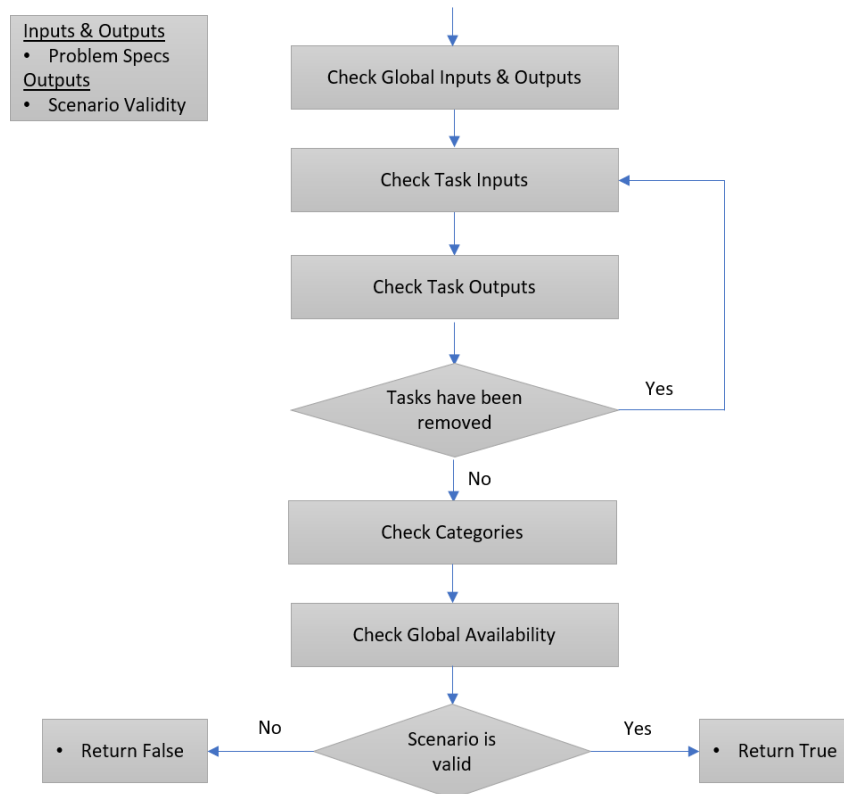


Figure 4.1 Pre-Processing algorithm

Each sub-process operation is performed only on the *remaining valid tasks* of the library of tasks. The valid tasks come from a Boolean array with size equal to the library size which in turn constitutes a record of the dictionary holding the problem specification and is updated during pre-processing. At the beginning, all the tasks in library are considered as valid. The aim of each sub-process is to search the remaining valid tasks for inconsistencies regarding the operation it performs, and if there are indeed inconsistent tasks, to make them *invalid* for the following sub-processes and the main optimization procedure. At the end, a Boolean variable is returned to signify if at least one feasible design can be composed by the remaining valid tasks according to the process requirements and, by extension, whether the main optimization operation should be performed or not. In the following sub-sections, the sub-processes will be presented along with their pseudocode.

4.2.1 Check Global Inputs & Outputs

The first sub-process searches for inconsistencies between the resources of the valid tasks in library and the process requirements. Specifically, this sub-process is the merger product of two of the discoveries made in the previous section. As it shown in the pseudocode of this sub-process in figure 4.2, the merging of those two discoveries assists in code efficiency.

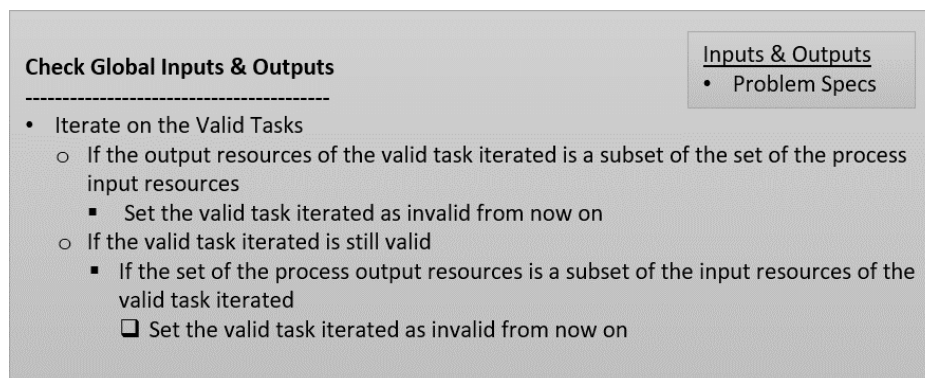


Figure 4.2 Pseudocode of “Check Global Inputs & Outputs” sub-process

The first discovery has to do with the tasks whose set of output resources is a subset of the set of process input resources. These tasks cannot be part of an optimal process design, not even of a feasible process design regarding the problem specification, and perhaps they could be part of a business process that precedes the examined process. The second discovery concerns the tasks whose set of input resources is a superset of the set of the process output resources. Again, such tasks cannot be part of a process design regarding the examined process specification and they could probably be part of a process that follows the examined one. Therefore, the responsibility of this sub-process is to search for such tasks and if any, to make them invalid for the optimization phase.

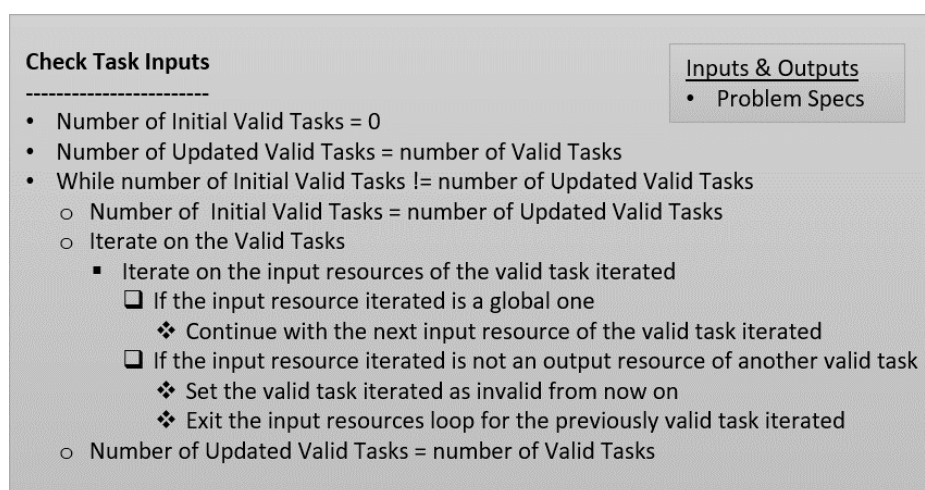


Figure 4.3 Pseudocode of “Check Task Inputs” sub-process

4.2.2 Check Task Inputs

This sub-process seeks for tasks that at least one of their input resources can be obtained neither by the output resources of other tasks nor by the process input resources. In this operation, the valid tasks are considered only, and if such tasks turn out to be problematic, they will be made invalid. The pseudocode of this sub-process is presented in figure 4.3 above. As it shown in the pseudocode above, the sub-process is repeated until no more valid tasks become invalid. The need for this repetition is that if a valid task becomes invalid, there may be some other formerly valid tasks, whose some of their input resources could only be obtained by the output resources of that currently invalid task, thus these must become invalid too.

4.2.3 Check Task Outputs

This sub-process seeks for tasks that all their output resources cannot serve as input resources for other tasks or as process output resources. In this operation, the valid tasks are considered only, and if such tasks turn out to be problematic, they will be made invalid. The pseudocode of this sub-process is presented in figure 4.4 below.

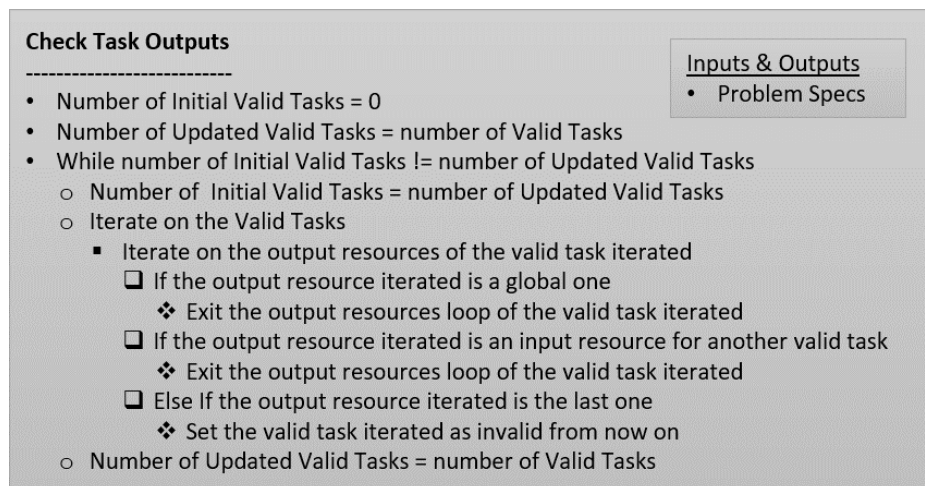


Figure 4.4 Pseudocode of “Check Task Outputs” sub-process

The pseudocode above, contains a repetition of the sub-process like the one from the “Check Task Inputs” sub-process for a similar reason. If a valid task becomes invalid, there may be some other formerly valid tasks, whose all their output resources could only be given to satisfy the input resources of that currently invalid task, thus these must become invalid too. However, there is a difference between the “Check Task Inputs” and “Check Task Outputs” sub-processes. For the first one, one unsatisfied input resource of a task is enough for this task to become invalid. On the other hand, for the second one, all the output resources of a task cannot be given anywhere in order to become invalid. As discussed in previous sub-section, the former case demonstrates a hard constraint for the BPO problems, “All the input resources of task must be available before its execution” but the latter case has to do with needless tasks absolutely.

4.2.4 Check Categories

It is the author's opinion that this sub-process is the most important because its aim is to remove the inefficient tasks from the library. The inefficiency is inferred by the fact that every business process design that contains any of these tasks is always a sub-optimal one. For example, let's say that the examined problem is a bi-objective one and its problem type is "MIN-MIN" so the desideratum is to minimize both the process attributes. Imagine that a process design has been captured by the framework that contains, among others, a specific task whose attribute values are (10, 10). Now, there is another task in the library, whose sets of input and output resources are identical with those of that task; its attribute values are (9, 9), though. If we change the former task with the latter one in the produced process design, the new process design will always be better than the previous one since the contribution of the new task to the process attribute values will be less. So, if the first task is removed from the library, the probability of the second task to be part of a process design will increase leading to better solutions. As someone can infer from the pseudocode in figure 4.5, this sub-process comprises two steps. The first step is to find the categories that the valid tasks of the library can be classified in according to their sets of input and output resources. The tasks of a specific category have identical input and output resources. The second step is to traverse the tasks of those categories and find the tasks which are dominated by other tasks of the same category considering the problem type and if any, these tasks must become invalid.

Check Categories

- Initialize Categories as an empty dictionary
- Mark all Valid Tasks as not belonging to a specific category yet
- Iterate on Valid Tasks
 - If the valid task iterated does not belong to a specific category
 - Mark the valid task iterated as belonging to a category
 - Add a new category to the Categories dictionary
 - Add the valid task iterated to the newly created category
 - Iterate on the rest Valid Tasks
 - If the inner valid task iterated does not belong to a specific category
 - ❖ If the outer valid task iterated has exactly the same input and output resources with the inner valid task iterated
 - Mark the inner valid task iterated as belonging to a category
 - Add the inner valid task iterated to the newly created category

- Iterate on Categories
- If the category iterated has more than one Valid Tasks
 - Get the Problem Type e.g. ("MIN-MIN", "MIN-MAX", "MAX-MIN", "MAX-MAX")
 - Iterate on the tasks of the category iterated
 - If the category task iterated is still valid
 - ❖ Iterate on the rest tasks of the category iterated
 - If the inner category task iterated is still valid
 - ✓ If the inner category task iterated dominates the outer category task iterated according to the Problem Type
 - @ Set the outer category task iterated as invalid from now on
 - @ Exit the loop on the rest tasks of the category iterated
 - ✓ Else If the outer category task iterated dominates the inner category task iterated according to the Problem Type
 - @ Set the inner category task iterated as invalid from now on

Inputs & Outputs

- Problem Specs

Figure 4.5 Pseudocode of "Check Categories" sub-process

4.2.5 Check Global Availability

The last sub-process of the pre-processing stage performs the basic operation of examining if the given library of tasks is appropriate for the configured process requirements. As it was described in the previous sub-sections, the outcome of the rest sub-processes is the update of the validity of the tasks in the library. After pre-processing, only the valid tasks will be considered during the optimization phase. Furthermore, two of the requirements for a business process design to be feasible are: 1) At least one of the process input resources must be utilized during the process by the participating tasks and 2) all process output resources must be produced during the process by the participating tasks. Hence, it would be valuable for the optimization procedure and the assessment of its results to know beforehand if there were tasks in the library that could meet the process requirements. In addition, the process requirements are configurable, and it's the user's responsibility to provide the proper ones. In case of wrong configuration, especially when the framework is used as a process mining tool for a given library, there must be a way to prevent the framework from producing misleading results and redundant executions. Moreover, in such cases no process designs will be produced, and the wrong configuration should be easily distinguished from the complex or infeasible problem. Figure 4.6 demonstrates the pseudocode of this sub-process. At the end of this sub-process, a Boolean variable is returned to signify whether the optimization phase should be executed or not.

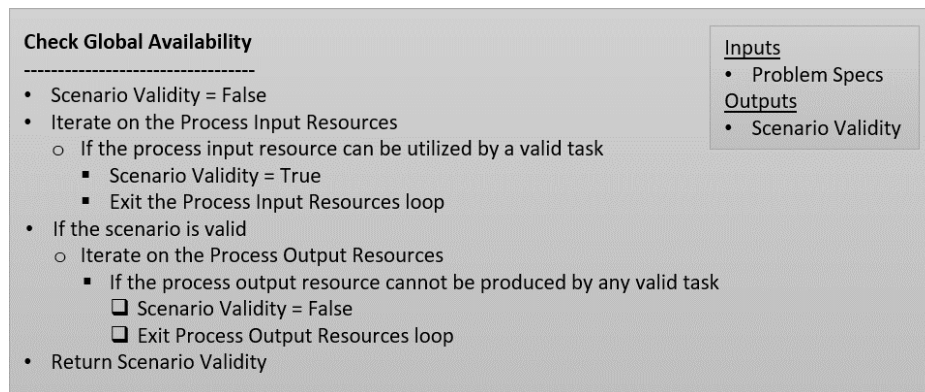


Figure 4.6 Pseudocode of "Check Global Availability"

4.3 Summary

This chapter discussed the purpose of integrating a pre-processing stage in business process optimization and introduced the proposed pre-processing technique for business process optimization problems. This technique aims for increasing the ability of the new optimization framework to find the optimized design alternatives by removing the tasks that would never be part of such a design. The next chapter introduces the proposed process composition algorithm that extends the scope of the framework towards complex real-life problems.

CHAPTER 5

Process Composition Algorithm (PCA-II)

This chapter introduces the new *process composition algorithm* proposed in this thesis. It starts by discussing the operation that such an algorithm performs within the optimization framework. Next, the weakness of the former algorithm to deal with more-real life scenarios is discussed that led the author to develop PCA-II, the new process composition algorithm. PCA-II constitutes the successor of PCA which was developed by Vergidis (2008). The chapter concludes with the presentation and elaboration of the main steps of PCA-II along with its new features and the considerations made during the development of these steps.

5.1 Process Composition

The process composition is the step where the quantitative representation of a business process is transformed to the visual one. In the proposed optimization framework, this step is part of the evaluation phase that comes after the genetic operators have changed the solutions in the population and before the evolution of this population takes place. Figure 5.1 demonstrates these steps in a visual manner.

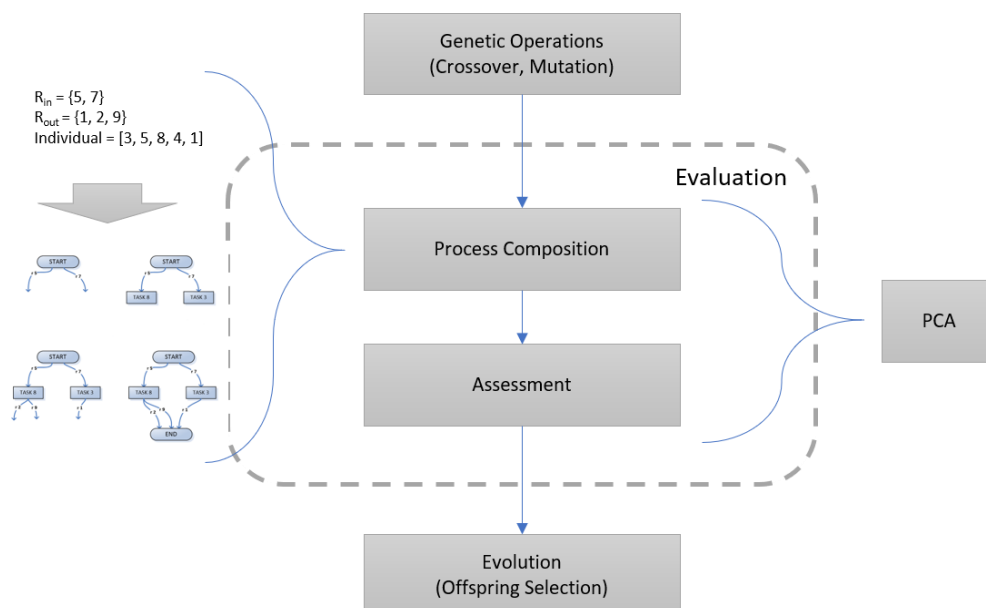


Figure 5.1 Evaluation Phase

During evaluation, the process composition algorithm attempts to compose a feasible process diagram for every solution in the population using its quantitative representation and considering the process requirements. As it shown in figure 5.1 above, the process composition algorithm is also responsible for the assessment of every solution in the population in terms of examining whether the produced process design for that solution is feasible according to the problem specification or not, and properly updating its attributes values that will be used in the evolution of the population afterwards.

5.2 Necessity of PCA-II

The aim of the business optimization framework is to capture, visualize and express a business process design in a quantitative way that allows EMOAs to generate a series of alternative optimized designs, Vergidis (2008). However, the procedure of composing at least a feasible process design isn't trivial because of the constraints that this must satisfy. As Vergidis pointed out, the feasibility constraints yield a significant number of infeasible cases and, as such, the probability of infeasible designs composed by the process composition algorithm is very high. Therefore, such an algorithm should be able to acknowledge such infeasibility issues and have a strategy for surpassing them if it's possible.

5.2.1 Low quality Results

In the previous chapter, the pre-processing of the library of tasks has been presented. The validation testing of that phase has been made with the real-life problems that were used by Vergidis (2008) to validate BPO_F, the proposed optimization framework in his research. The validation of the pre-processing phase is successful, and the results will be presented in a following chapter. However, the results of the execution of BPO_F for the same problems after pre-processing, were unexpectedly disappointing. The pareto front of the solutions wasn't even the same with that without pre-processing, but it was always worse in all runs conducted. Additionally, after pre-processing on the library of the problem "Scenario B: Sales forecasting", BPO_F was unable to find a whole island of solutions. The term, island of solutions, refers to all solutions comprising a specific number of tasks. Figure 5.2 depicts the results of BPO_F for the Scenario

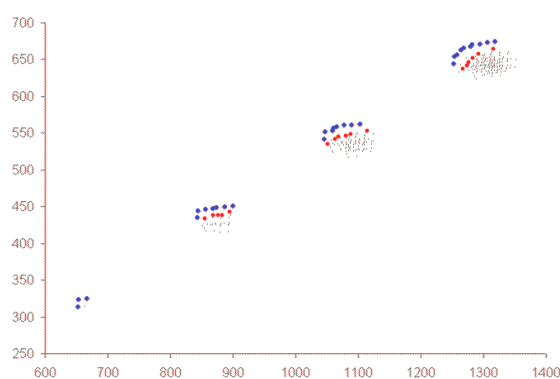


Figure 5.2 Scenario B with and without Pre-Processing for BPO_F

B. Blue dots correspond to the pareto front without pre-processing and the red ones correspond to the pareto front with pre-processing. At first glance, someone could say that the pre-processing

makes the things worse for the evolutionary algorithms. However, this is not the case and as it will be discussed in the following section, those inconsistencies stem from the approach of PCA.

5.2.2 Weakness of PCA

The pre-processing algorithm has been designed by the author of this thesis with the intention to be generic enough to be applicable to any problem dataset. Therefore, the considered constraints for the feasibility of a business process design should be well-rounded. On the other hand, the problems that Vergidis (2008) was involved with, came from the service industry and the proposed optimization framework in his research was designed considering the needs of that domain. According to those needs, Vergidis considered the following rules, when designing PCA, for the feasibility of the produced business process designs:

1. All process input resources are utilized by one or more tasks participating in the process design
2. All process output resources are produced by one or more tasks participating in the process design
3. Each task in the design is connected either with the process inputs, the process outputs or another task in the design

The first two constraints are pretty straightforward and stem directly from the requirements of the examined business process itself. The third constraint implies that every participating task in the process design, must utilize at least one input resource from the process input resources or the output resources of another participating task and give at least one of its output resources to another participating task or to produce at least one of the process output resources. This rule can ensure the feasibility of processes whose tasks require only one input resource, but generally the participating tasks in a process design may need more than one input resources to be executed. An equivalent rule for the relationship between the activities and the resources by Hofacker and Vetschera (2001) assume that *all input resources of an activity must be available before this can be executed* and thus, *for all activities contained in the process design, all their input resources are either satisfied by the process input resources or are generated by some preceding activity*. This is also the author's view and, as such, one of the infeasibility cases that the pre-processing algorithm looks for. Hence, the root cause of the inconsistencies of the produced results was the inconsistency between pre-processing and PCA on what makes a process design perceived as feasible.

The dataset of "Scenario B" is presented in figure 5.3 and an optimized solution of the missing island is shown in figure 5.4. As presented in chapter 6, a step of pre-processing involves the removal of tasks whose at least one of their input resources can be obtained neither by the output resources of other tasks nor by the process input resources. The **tasks 0, 3, 9, 11 and 12** need the

resource 1 but no other task or process input resource can satisfy this demand. So, these tasks cannot be part of a feasible design and must be removed.

No.	Task Name	Input(s)	Output(s)	SDP	SFT
0	D&B Business Verification	3,1	0,5	206	103
1	Fax.com	8,2	4	220	103
2	Gale Group Business Information	3,0	5	223	106
3	Gale Group Business Intelligence	3,1	0,5	229	113
4	GraphMagic's Graph & Chart Web Service API	5,8	2	203	107
5	interfax.net	8,2	4	222	113
6	Lokad Business time-series forecasting and analysis	5,7	8	228	110
7	Midnight Trader Financial News	6,3	7	217	101
8	Strikelron Company Search	3	3,0	230	114
9	Strikelron Get Business Prospect	3,1	0,5	205	110
10	Strikelron Lookup Business	3	3,0	201	110
11	Wall Street Horizon Real-Time Company Earnings	3,1	5	210	105
12	Xignite Get Balance Sheet	3,1	5	216	112
13	Xignite Get Chart Url	8	2	228	110
14	Xignite Get Chart Url Preset	8	2	228	101
15	Xignite Get Growth Probability	5,7	8	215	109
16	Xignite Get Market News Headlines	6	7	221	114
17	Xignite Get Market Summary	6	7	203	112
18	Xignite Get Topic Chart	3,5,7	8	218	112
19	Xignite Get Topic Data	3,5,7	8,2	222	109

Task library of scenario B

No.	Resource name	Process I/O
0	Business details	
1	Business query	
2	Chart / graph	
3	Company name	I
4	Fax (on-line)	O
5	Financial data	
6	Market update request	I
7	Recent market trends	
8	Time-series forecast	

Available Resources for scenario B

Figure 5.3 Scenario B by Vergidis (2008)

On the contrary, the **task 12**, “Xignite Get Balance Sheet”, is part of a missing solution composed by PCA without pre-processing, even though it can’t get **resource 1**. Additionally, the **task 5** “interfax.net” requires the **resources 8 and 2** to be executed but it can get only the **resource 8** from the participating tasks.

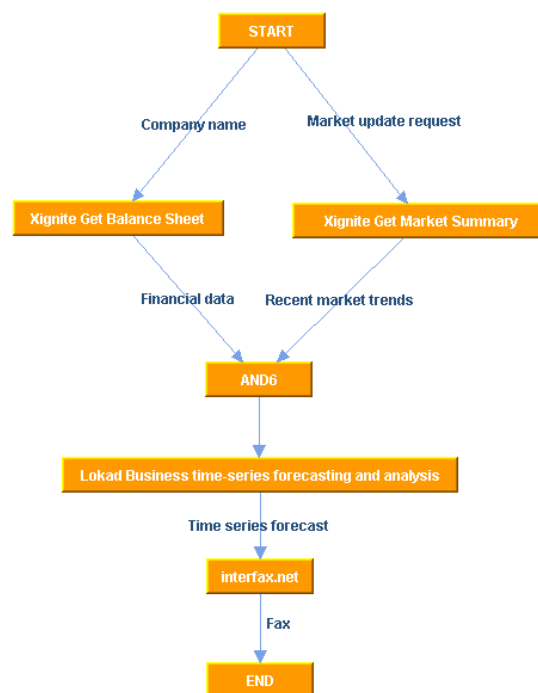


Figure 5.4 A 4-task optimized process design by Vergidis (2008)

The answer to those contradictions is found to the steps of PCA. As it shown in figure 5.6, the “ELABORATE CHILD LEVEL” operation is responsible for attaching new tasks to the parent level tasks. The criterion for a new task to be attached, is to utilize the output resource iterated of the parent task iterated. These tasks will constitute the next parent level if all process output resources won’t be produced. It’s obvious that this approach doesn’t ensure that the next parent tasks can get all their input resources, and this is exactly the reason why PCA isn’t applicable to more complex problems. This algorithm couldn’t have a step involved with the satisfaction of the

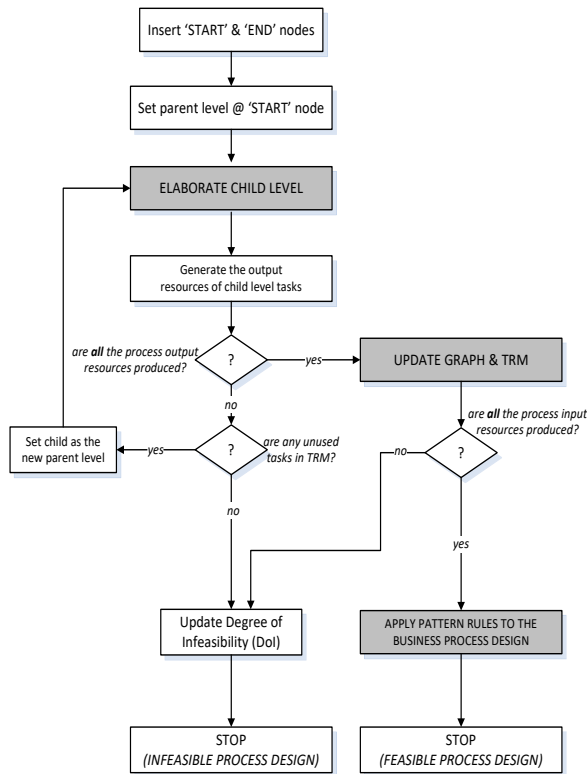


Figure 5.5

Main steps of PCA by Vergidis (2008)

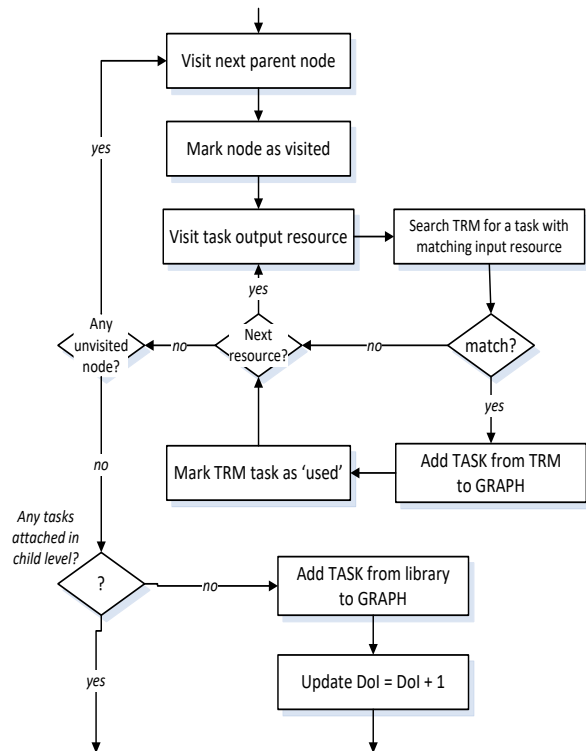


Figure 5.6

“ELABORATE CHILD LEVEL” operation by Vergidis (2008)

input resources of the participating tasks because of it’s *top-down* composition approach. Thus, a new process composition algorithm should be developed in order for the optimization framework to handle more complex problems.

5.3 PCA-II

This section introduces PCA-II, the proposed process composition algorithm for replacing PCA in the optimization framework. This algorithm aims to capture the needs of a wide range of business process domains so that the framework will be capable of processes beyond the service industry. The section starts with the assumptions of the author regarding the feasibility of a generic business process and the approach which the new algorithm follows. Next, the business process quantitative representation will be presented along with the inputs and the outputs of the

new algorithm. Finally, the main steps of PCA-II and their aspects are presented and discussed in detail.

5.3.1 Infeasibility of process designs

The assumptions considered by the author for the feasibility of a business process design, derive from the two business process optimization approaches which this research is based on. Both approaches combined BPO with EMOAs, which is also the scope of this thesis. The first approach modelled the business process in a more generic way and is owed to Hofacker and Vetschera (2001). The second approach comes from Vergidis (2008) who developed the first optimization framework for problems of the service-industry. According to the author of this thesis, a business process design is feasible if the following constraints are satisfied by the participating tasks:

1. All process output resources are produced by one or more participating tasks in the design
2. All input resources of the participating tasks in the process design are satisfied either by the output resources of their preceding tasks in the design or by the process input resources
3. All participating tasks in the process design provide at least one of their output resources for satisfying a process output resource or an input resource of a succeeding task in the design

The first and the third rules are applied in both approaches and the second one is considered only in the second approach. It's the author's opinion that these rules are generic enough to ensure the feasibility of any process design. Figure 5.7 depicts a feasible process design.

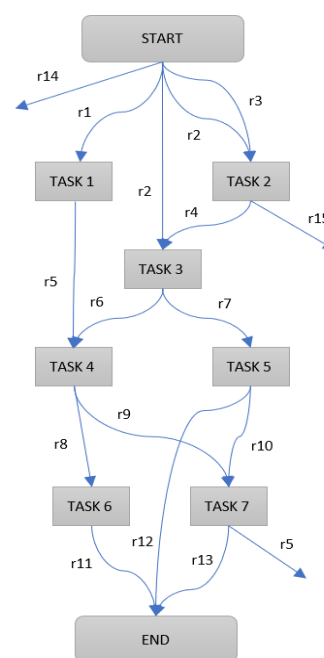


Figure 5.7 A feasible process design

The corresponding infeasibility cases to the constraints above, are:

1. A process output resource cannot be produced by any task in the examined solution
2. The input of at least one participating task cannot be satisfied from the tasks in the examined solution or the process input resources

In these cases, PCA-II stops *an attempt* to compose a feasible process design. This behaviour will get clearer in the section where the main steps of PCA-II are presented. Someone may wonder why no infeasibility case occurs from the third constraint above, but next, the composition approach of PCA-II is presented, which ensures that the third constraint is always satisfied.

5.3.2 Composition approach

PCA-II follows the *bottom-up* composition approach which is exactly the opposite of PCA. It starts with the satisfaction of the process outputs resources and continues with the satisfaction of the input resources of the participating tasks in design. For a feasible process design, the operation stops when all participating tasks have their input resources available by the process inputs resources or by the output resources of their *preceding* tasks in the design. The preceding tasks are actually *succeeding* within the process composition because the criterion for a new task to be attached during graph elaboration, is to satisfy an input resource of the parent task iterated or a process output resource with one of its output resources. This behaviour is demonstrated in the highlighted area in figure 5.8 and is the core difference between the approaches of PCA and PCA-II. When all new attached tasks can satisfy their input resources from the process input resources directly or by previously attached tasks, the process is considered as feasible and the operation stops.

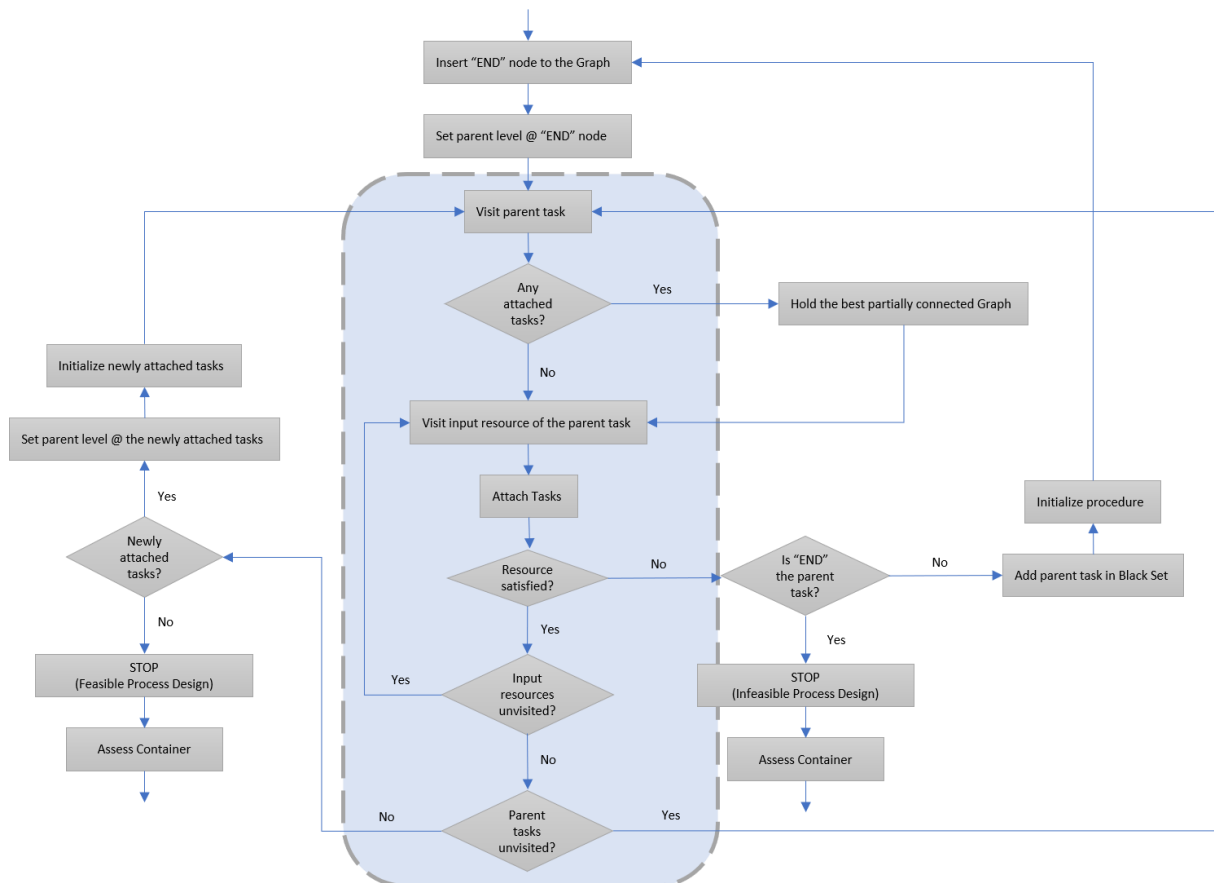


Figure 5.8 Composition approach of PCA-II

For the infeasibility cases during elaboration, when a new task cannot satisfy one of its input resources, it's added to the “*Black Set*” and a *new process composition attempt starts from scratch*. The black set contains the ineligible tasks for attachment in future composition attempts. When the “problematic” task is the “END” node, the operation stops, and the solution is considered as

infeasible. The third constraint of the previous section for the feasibility of a process design is always satisfied by the operation itself.

5.3.3 Business process representation

The evolutionary optimization algorithms maintain a population of possible solutions of the problem examined and evolve it for a predefined number of generations in a way that at the end, this population will contain the optimal solutions. In accordance with this feature of EMOAs, PCA assesses the feasibility of the solutions in the population in a way that at the end, the population will contain the alternative most optimized process designs for the process requirements. Every solution in the population starts with the maximum numbers of tasks that a solution can have. Afterwards, through generations, PCA modifies the solutions by adding or removing tasks. Finally, every feasible solution contains only the participating tasks.

On the other hand, eBPO_F considers every *solution* in the population as a *container of tasks* and PCA-II *makes attempts* to compose a feasible process design using some of those tasks. The assessment of the containers involves the ability of PCA-II to compose a feasible process design with their tasks and the attribute values of those tasks. At the end, the population must contain the *most promising* containers for composing a feasible optimized process design and this is the aim of the evaluation performed in the new algorithm. At the whole optimization phase, all containers in the population, have the same predefined number of tasks and no change is performed to the tasks of a container except for the genetic operators, crossover and mutation.

Through generations, PCA-II examines every container in the population whether it contains a feasible process design or not. In both cases, after the composition attempts, a *list with the best partially connected tasks* in the design and the *graph* of this design will be used for the assessment of the examined container. In case of process feasibility, the graph corresponds to the *first feasible process design* composed by the algorithm and the list contains all participating tasks. Otherwise, the graph corresponds to the best possible elaboration of the design with the tasks in the container before one of those tasks cannot be satisfied for one of its input resources and the list contains the participating tasks in that design.

The tasks in the list mentioned above, are added in the same order which are attached to the graph during the composition attempt. This list and the graph for a feasible design are added in “*Hall of Fame*” where all feasible designs found in the execution of eBPO_F, are stored. In addition, the order which the tasks are presented in the container, matters; the searching for a task in the container that satisfies the examined input resource, is always performed from left to right. In this way, PCA-II ensures that:

1. Given a container of tasks, it will always construct the same “most feasible” design during its composition attempts
2. The results of eBPO_F execution can be reviewed. The process composition of the tasks of the list-container accompanying a feasible solution leads to the same process design produced by eBPO_F

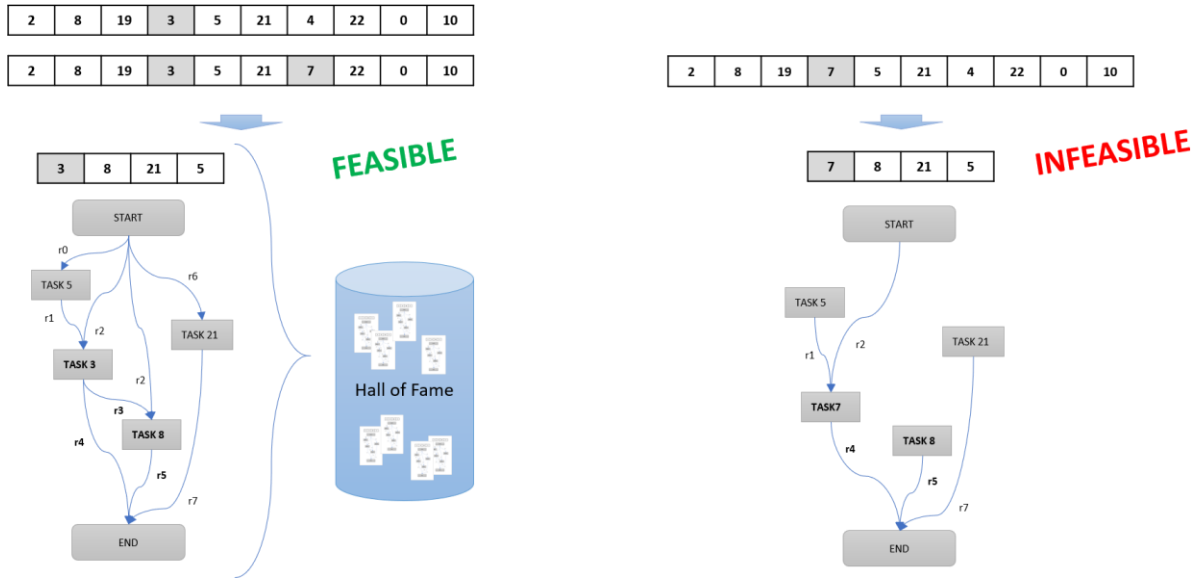


Figure 5.9 Composition outcomes of PCA-II

Figure 5.9 shows the results of the process composition phase for four similar containers in the population. These will be used for the assessment of each container, respectively. All designs correspond to the most elaborated designs found within the composition attempts. In the infeasible design for the container in the right, the **task 8** cannot satisfy its need for **resource 3**, there is no other tasks to satisfy the **process output resource 5**, so the container doesn't contain a feasible design. On the other hand, the two containers in the left, have **task 3** which can satisfy the needs of the **process output resource 4** and the **input resource 3** of **task 8**. It's worth mentioning that the second container has also the **task 7** apart from the **task 3**, but the outcome is the same for both containers because the task 3 precedes task 7 and thus, it is attached first in the design for satisfying the process output resource 4 and afterwards, it can also satisfy the input resource 3 of the task 8 and no new task is attached.

However, this is not the case for the designs of figure 5.10. Both containers have the **task 3** and the **task 6** which have the **same input and output resources**. The only difference between those containers is the **order of tasks** in them. In the first container the task 3 precedes task 6 and in attempt to satisfy the **process output resource 4** the task 3 is attached first. In the second container the task 6 precedes task 3, hence the process design.

Similarly, the containers of figure 5.11 have the same set of tasks but in different order. The first design corresponds to the second container in the left of figure 5.9. The only difference in the second container of figure 5.11 is that the **task 7** precedes the **task 3**. First, the task 7 is attached

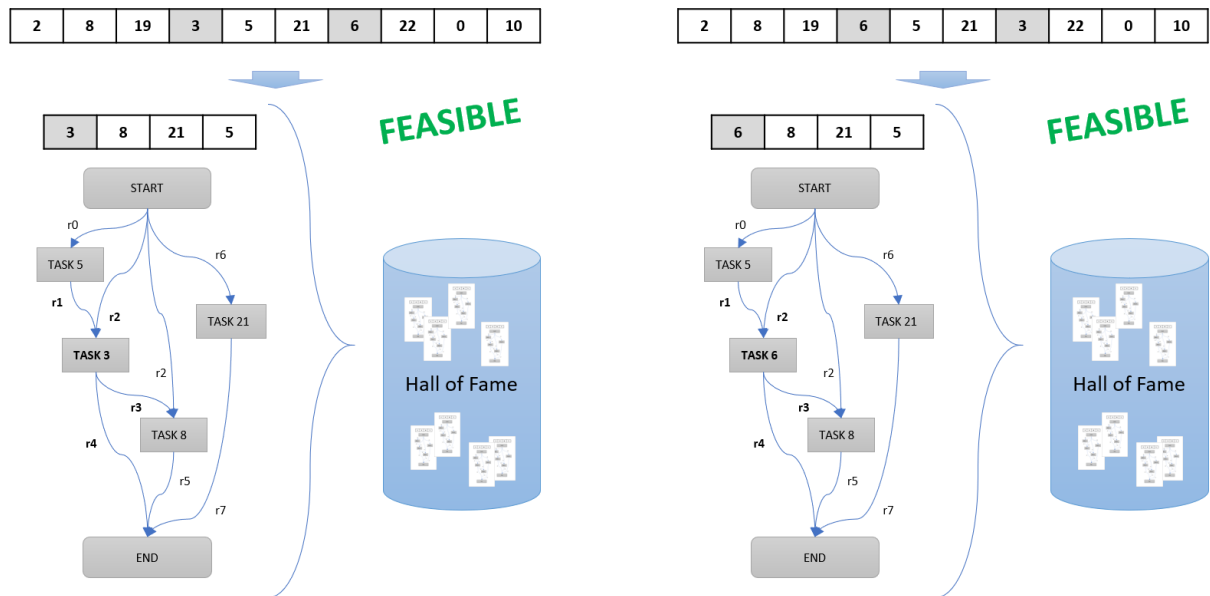


Figure 5.10 Same tasks in different order

to satisfy the **process output resource 4** and then the task 3 is attached to satisfy the **input resources 3** of the **task 8**.

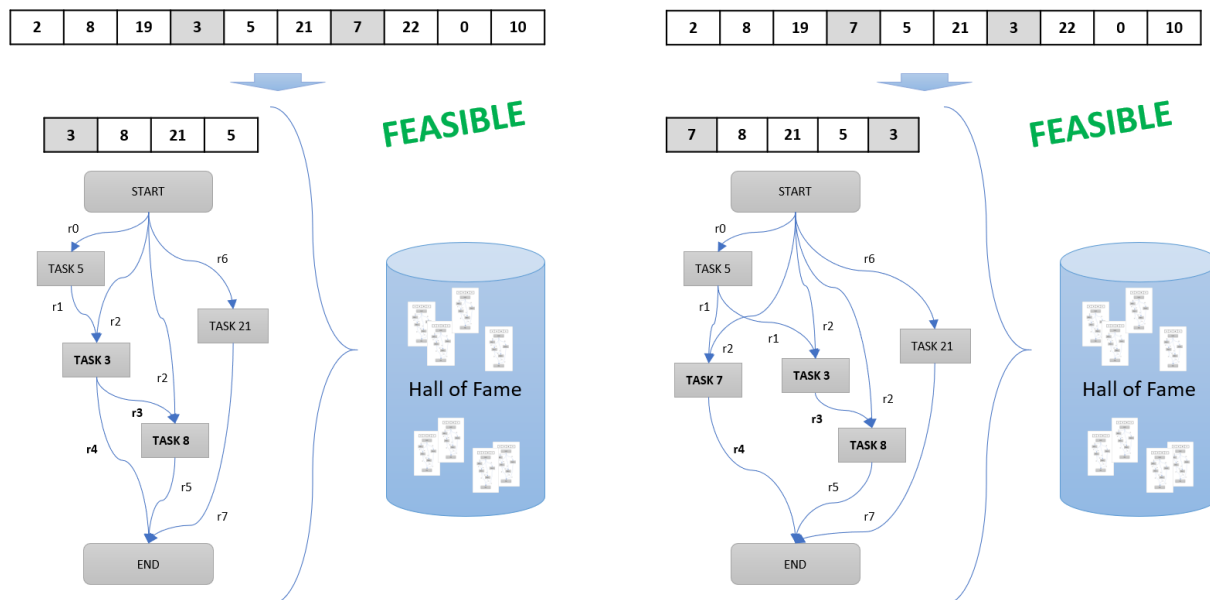


Figure 5.11 Tasks in different order

5.3.4 Business process assessment

After the process composition attempts, each container must be assessed in a way that the evolutionary optimization algorithm will select the most promising containers to be the parents for the next generation. In BPO_F, the selection of the solutions was made according to their *process attribute values*. In that phase, only a predefined number of solutions having the least degree of

infeasibility (DoI), took part. DoI was a measure suggested by Vergidis (2008) to measure the extent to which a process design is infeasible and helps in selecting the “less” infeasible solutions and preserving them in the population with the hope that they have a better chance of evolving towards feasible solutions during the optimization process.

DoI is also adopted in eBPO_F and plays a significant role in the optimization process. It is considered as an *extra attribute of the containers* along with the *container attribute values* and an extra objective is added to every business process optimization problem. The extra objective always involves the *minimization of DoI*. In eBPO_F, DoI is used for measuring the extent to which the most elaborated design found in the process composition phase, is infeasible. By assigning DoI to a container as an attribute, DoI reflects the ability of PCA-II to compose a feasible process design by the tasks of that container. Every container in the population which contain a feasible process design, has DoI equal to zero.

Additionally, instead of considering only the attribute values of the participating tasks, the attribute values of all tasks in the container, are considered in eBPO_F. These attributes values constitute the *container attribute values*. This assessment approach owed to the scope of eBPO_F to produce the optimized design alternatives regardless the size of those designs and is adopted to lead the evolution phase towards the most promising containers. The containers of tasks and the composition approach followed in eBPO_F, provide the ability to find feasible optimized designs of different size. Therefore, the consideration of the container attribute values instead of the process attribute values of BPO_F, helps in situations where two containers contain feasible designs of different size. Considering only the participating tasks, the container with the smallest design would always be selected. Furthermore, considering the container attribute values along with DoI, the extra objective, containers which cannot provide a feasible design, but their tasks have the best attribute values, will have a disadvantage against containers with “worse” tasks which contain a feasible process design. In case, two containers provide feasible designs regardless size, the selection is in favour of the container with the lowest container attribute values since DoI equals zero for both, following the notion that *the container with the best container attribute values will be more promising for evolving towards a container with a more optimized feasible design*.

DoI is based on three main factors for the infeasible containers and one factor for the feasible ones. The terms “feasible” and “infeasible” containers refer to the ability of PCA-II to construct a feasible design with their tasks. During process assessment, the container is assessed against the feasibility constraints that the most elaborated design violates and, a different weight is assigned for each constraint. These weights are selected in a way that reflects the relative importance and the frequency of each infeasibility case as well as the chance of those issues to be fixed through generations. Thus, DoI is calculated gradually. At the beginning of process assessment, DoI

equals zero and then, a different penalty is added for each constraint that violates. In this way, the more constraints a container violates, the less promising will be for next generations.

For an infeasible container, if its tasks cannot even satisfy the process outputs, the container size will be added to DoI as it is considered the most important constraint for the feasibility of the design. If no participating task utilizes a process input, DoI is increased by the half of the container size. Finally, the maximum number of unparticipating tasks and participating ones, is added to DoI. In this way, the container that cannot even satisfy the process output resources is considered as the less promising and it has the biggest DoI. On the contrary, in case of an input resource not satisfied, which is considered the most common case, the container is assessed leniently according to its needs for being a more promising one. Furthermore, if any process input resources aren't utilized, this means that the tasks form a partially connected graph where there is no much room for improvement since the participating tasks cannot touch the process start.

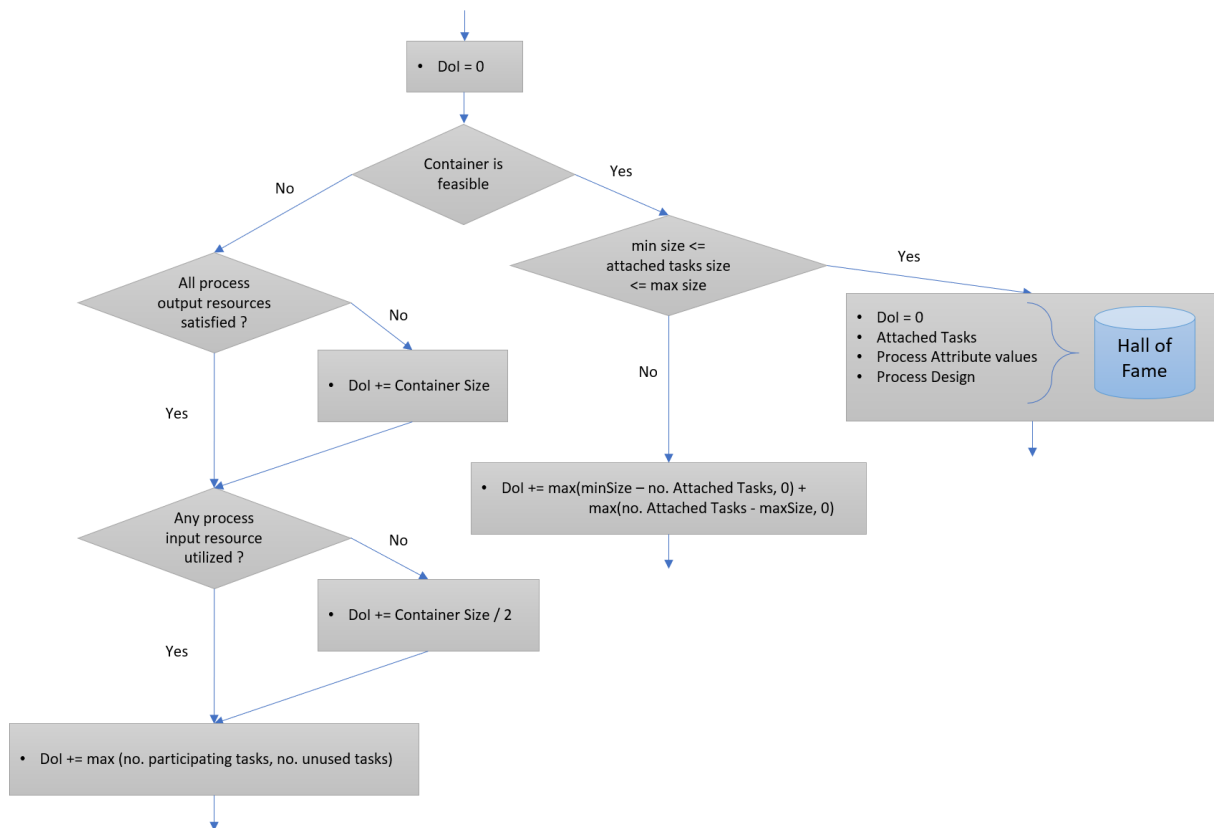


Figure 5.12 Calculation of DoI

For a feasible container normally, a new container is created only with the participating tasks in the design and the container attribute values which are equal to the process attributes values, considering only the participating tasks and DoI equal to zero. The newly created container and the process design will be added to Hall of Fame. Hall of Fame stores all feasible solutions found during the execution of eBPO_F and constitutes the outcome of this framework. The actual container has also DoI equal to zero in attempt to be preserved in the population. However, a

feasible container could have a larger DoI and wouldn't added in Hall of Fame, if the size of the process design violated the size limits set by the user. In such a case, the difference between the number of participating tasks and the violated limit, is added to DoI.

5.3.5 Main steps of PCA-II

Given a container of tasks and using some those tasks, PCA-II makes attempts to construct a process graph that meets the process requirements and complies with the feasibility constraints. In the graph, each task is represented by a node and there are two artificial nodes, the "START" node with the process input resources and the "END" node with the process output resources. These nodes facilitate the connection of the process input and output resources with the participating tasks. The resources are represented as directed arrows which start from the task providing the resource and end to the task receiving it. The graph is elaborated with the breadth-first strategy using the concepts of "parent" and "child" levels. The "parent" level comprises the tasks already inserted to the graph and the "child" level is the one where the new tasks are attached to the design based on the input resources of the tasks in the "parent" level. Once the elaboration of all tasks in "child" level is completed, it becomes "parent" level for the graph elaboration to proceed.

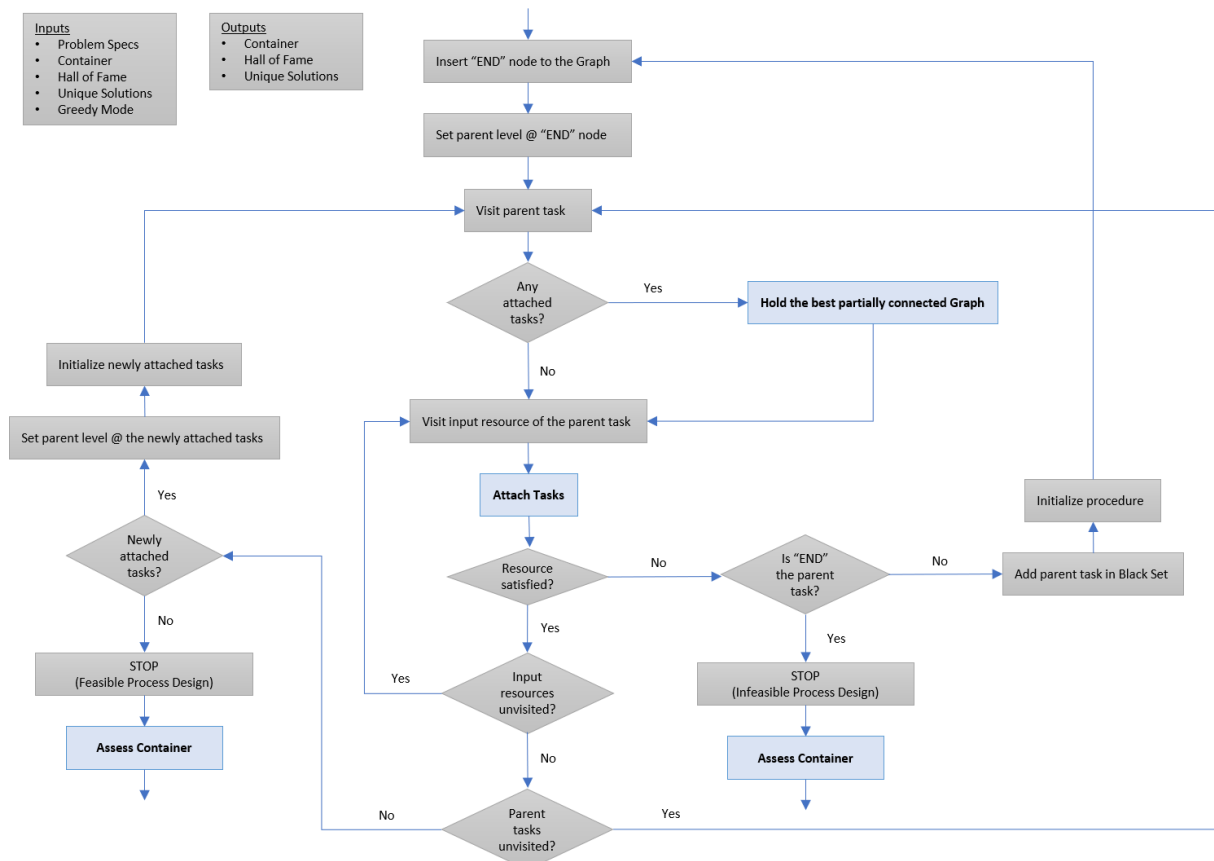


Figure 5.13 Main steps of PCA-II

Figure 5.13 depicts the main steps of PCA-II. It starts by inserting the artificial node “END” to an empty graph and to the parent level. Then, the algorithm visits all nodes in the parent level one by one in order to elaborate the child level. The graph elaboration is made by the “*Attach Tasks*” algorithm and involves the satisfaction of the input resources of the parent task iterated. If all input resources of the parent task iterated are satisfied, the *best partially connected graph* is updated, and the next parent node is visited. Once all parent nodes are satisfied, if there are newly attached tasks in the child level, these will form the next parent level nodes. Otherwise, the process design captured in the graph, is considered as feasible and the assessment of its container begins. At some point of time, if a parent task cannot satisfy one its input resources, this task is added to the *Black Set* a new process attempt starts from scratch without considering the tasks in Black Set. When more tasks are inserted in the Black Set during the attempts of PCA-II, less tasks remain to the container for attachment. At the end, the “END” node cannot satisfy the process output resources and no more composition attempts are made. In this case, the best partially connected graph found during attempts, is marked as infeasible and is used for the assessment of its container afterwards.

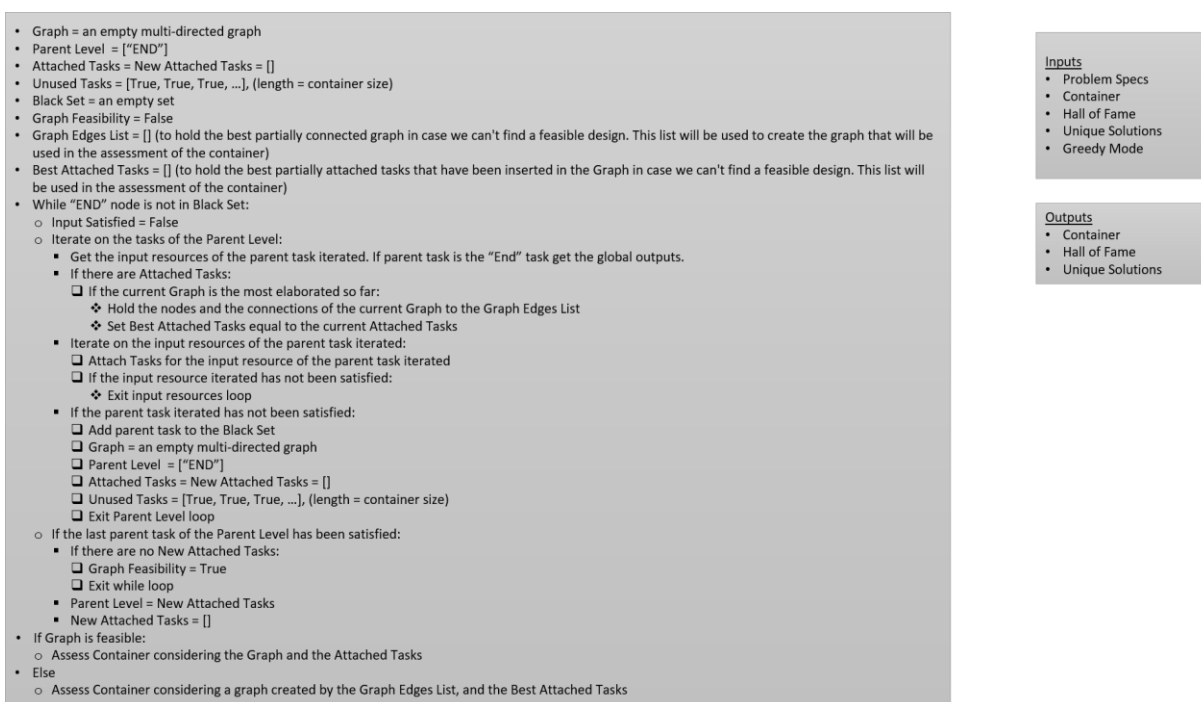


Figure 5.14 Pseudocode of PCA-II

5.3.5.1 Attach Tasks

Figure 5.15 shows the algorithm for the “Attach Tasks” operation. This operation is responsible for the graph elaboration. Given the input resource of the parent task iterated, the algorithm checks if that can be obtained directly from the process input resources. In such a case, the artificial “START” node is inserted to the Graph, the algorithm connects it with the parent task iterated for the examined input resource and returns a “True” value to indicate that the input resource has

been satisfied. In a different case, the algorithm searches into the already attached tasks to the graph to find the task that can satisfy the examined input resource. If such task exists and is neither the parent task nor a successor of the parent task, a connection will be established between that task and the parent task for the examined input resource and the iteration over the attached tasks, will stop. The attached task shouldn't be a successor of the examined task because the execution order of the participating tasks must be preserved. Therefore, the iteration over the attached tasks is made in reverse order. The most recent attached tasks are the last ones which probably are in the parent level too, hence it's more possible that no connection has been established between those tasks and the examined one yet. Next, if the input resource has been satisfied and the "Greedy Mode" is disabled, the algorithm returns a "True" value to indicate that the input resource has been satisfied. Otherwise, if no already attached task can provide the examined input resource, the algorithm searches into the unused tasks of the container. These tasks shouldn't be in the Black Set and be identical with the parent task examined; identical tasks can be in the same container due to the genetic operations that will be presented in the next chapter. In case of a valid unused task, this is inserted to the Graph, a connection is established with the examined task, is marked as used for the rest of the current composition attempt and is appended to the lists holding the attached tasks and the attached tasks of the current parent level. In addition, the iteration over the unused tasks, stops and the algorithm returns a "True" value to indicate that the input resource has been satisfied. In case that the algorithm cannot find a source to provide the examined input resource, it returns a "False" value to indicate that the input resource hasn't been satisfied.

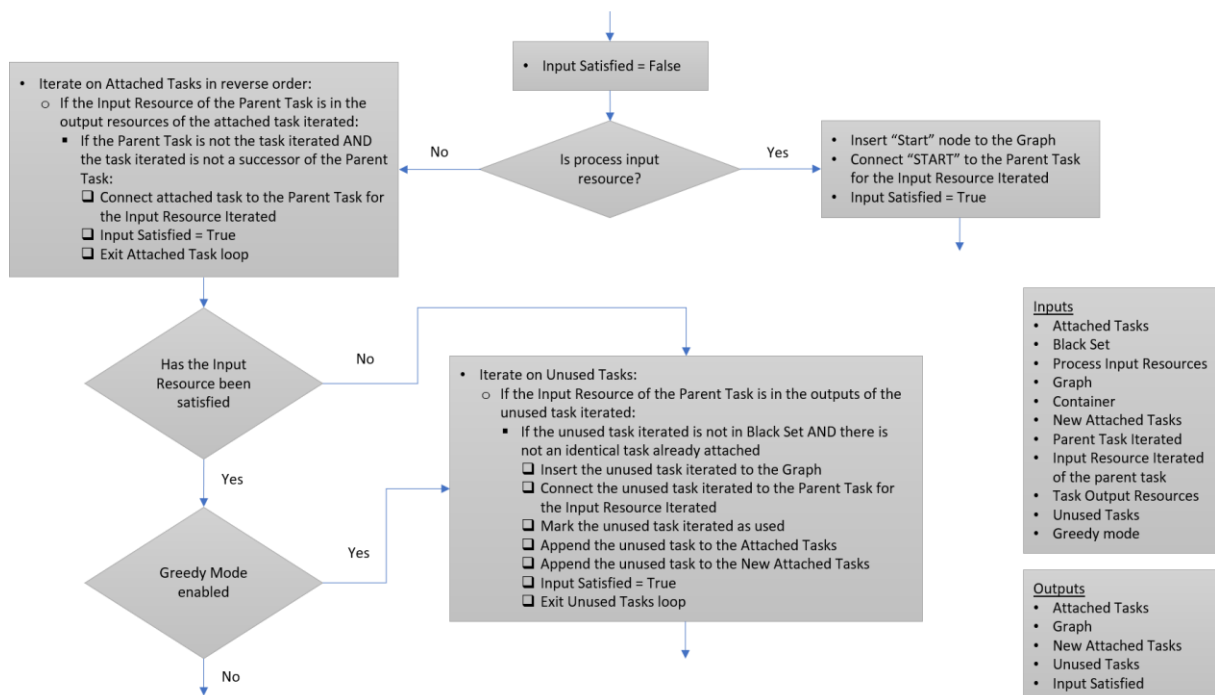


Figure 5.15 Algorithm of "Attach Tasks" operation

However, there is one more case where the input resource can be satisfied by an already attached task, but the algorithm will also search to find an additional unused task. This behaviour has to do with the “Greedy Mode” flag which is new feature of eBPO_F. The proposed optimization framework in this research maintains two populations of containers for the two different composition strategies followed by PCA-II. When Greedy Mode is disabled, PCA-II operates with the assumption of “Less tasks attached, less resources should be found”. Otherwise, PCA-II has the chance to construct more elaborated process designs and to explore even better the search space of the examined problem. This is the effect of the “OR” patterns formed when Greedy Mode is enabled. This is pretty useful in cases where a process design should involve back-up plans, OR patterns, among the participating tasks. For example, one of the objectives of such a process design might be the maximization of its reliability. Additionally, in this way PCA-II is also able to construct alternative feasible process designs with the same tasks. Figure 5.16 shows the results of PCA-II for two containers with the same tasks

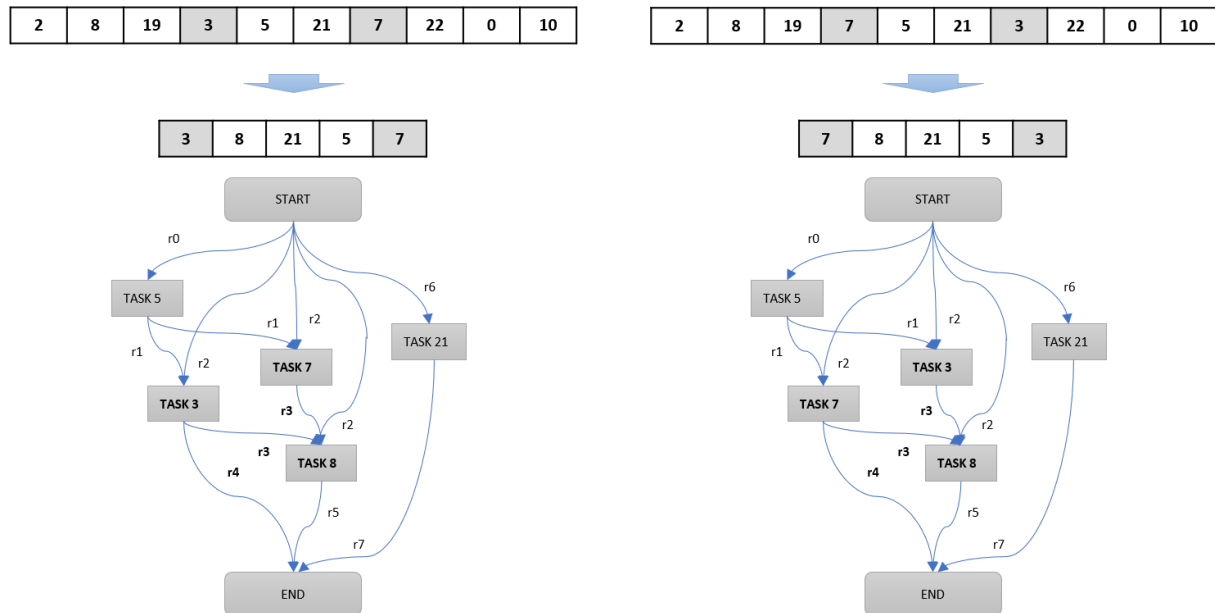


Figure 5.16 Two feasible designs elaborated with Greedy Mode enabled

when Greedy mode is enabled. The difference between these process designs is the utilization distribution of the resources across the involved tasks. The **tasks 3 and 7** have the same input and output resources. In the left design, the task 3 is attached first due to its position in the container and thus, it provides the **process output resource 4** and the **input resource 3** of the **task 8**. The task 7 only provides the input resource 3 of task 8. In the right design, those tasks have exactly the opposite responsibilities. It should be noticed, that both the process designs above, are the product of merging of two different 4-task solutions, $[3, 8, 21, 5]$ and $[7, 8, 21, 5]$, with an “OR” pattern for tasks 3 and 7.

5.3.5.2 Assess Container

The algorithm of “Assess Container” operation is presented in figure 5.17. At the beginning the DoI of the container is initialized to zero and then is updated according to the feasibility of the Graph extracted by the tasks of the examined container. DoI is an extra objective for eBPO_F and its calculation has been presented thoroughly in the previous section for the process assessment. The containers attributes values are calculated according to the objective function. eBPO_F assumes each container attribute value as the aggregate of the corresponding attributes of all tasks in the container. In case of a feasible process design, a new container with the participating tasks only and DoI equal to zero as well as its Graph, are added to Hall of Fame if it is the first time that this process design is found during optimization. Next, the list of the participating tasks is inserted to the “Unique Solutions Set” which is a data structure of eBPO_F used for the uniqueness examining. It should be noted that holding all feasible solutions found through generations in Hall of Fame for both populations instead of having only the feasible solutions of the last generation, as in BPO_F, allows us to have a better view of the search space of the examined problem.

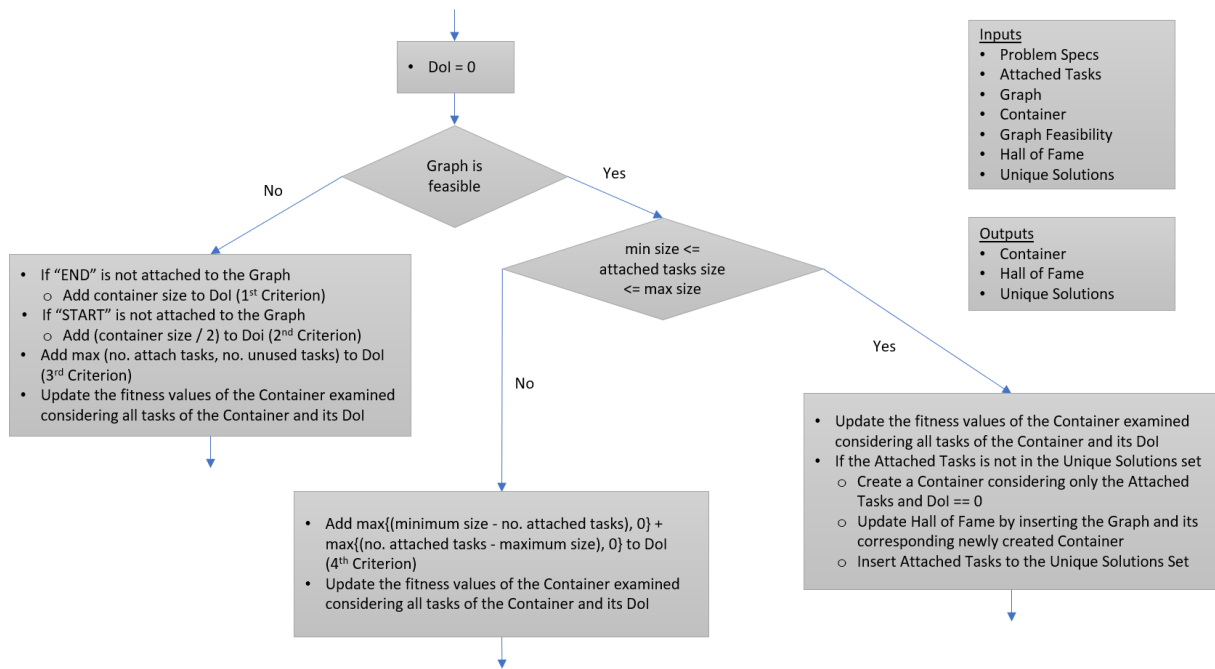


Figure 5.17 Algorithm of “Assess Container” operation

5.3.5.3 Hold the best partially connected Graph

The algorithm of holding the most elaborated Graph is very simple, as it is shown in figure 5.18 and involves the update of two data structures of eBPO_F, “Graph Edges List” and “Best Attached Tasks”. These structures will be fed to the assessment phase in case that the examined container cannot provide a feasible process design. The first structure holds the tasks and the connections of the most elaborated Graph and is used for constructing this Graph before the assessment phase. It is also used for examining whether the Graph after the next parent level elaboration is more

elaborated or not. The second structure just holds the participating tasks of the most elaborated Graph and is used for the calculation of DoI.

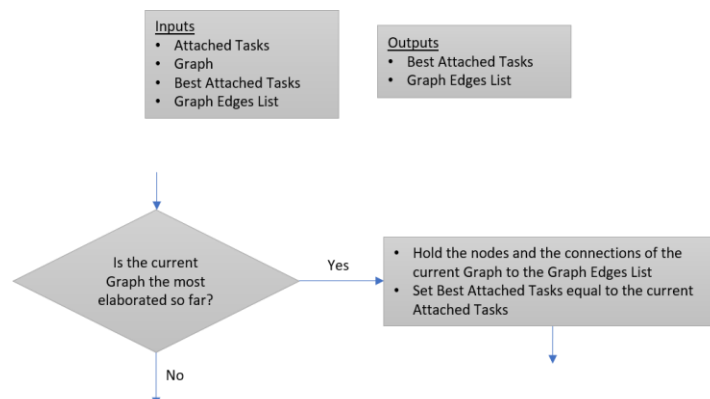


Figure 5.18 Alogrithm of “Hold the best partially connected Graph” operation

5.4 Summary

This chapter introduced PCA-II, the new process composition algorithm. This algorithm aims for capturing the design needs of a wide range of business process domains and enable the new optimization framework to handle intricate business processes. This chapter presented the context of process composition, the need for a new algorithm and the steps of this algorithm along with the considerations within the developing of those steps. The next chapter introduces the new evolutionary multi-objective optimization framework that uses PCA-II for process composition and evaluation and incorporates the proposed pre-processing technique, discussed in chapter 4.

CHAPTER 6

Extended Business Process Optimization Framework (eBPO_F)

This chapter introduces the *extended business process optimization framework* (eBPO_F) that has been developed within this research and is based on the existing framework implemented by Vergidis (2008). It starts by providing the mathematical formulation of the business process optimization problem and the functional overview: the inputs, the outputs and the main operation of the framework. Next, the evolutionary multi-objective optimization algorithms used in the new framework are presented, along with the solution representation during optimization. The chapter concludes with the implementation of eBPO_F, the refined implementation of the existing optimization framework in Python.

6.1 Problem Formulation

This section presents the formulation of the business process optimization problem. The problem formulation assumes that the business process design requirements are captured based on the proposed representation that has been discussed in previous chapter. Table 6.1 shows the problem parameters based on the proposed business process representation.

Parameter	Description	Parameter	Description
n	Number of tasks in the library	N	Set of the n tasks
n_d	Number of tasks in the design	N_d	Set of the n_d tasks
n_c	Number of tasks in the container	N_c	Set of the n_c tasks
n_{min}	Minimum number of tasks in the design	n_{max}	Maximum number of tasks in the design
r	Number of available resources	DoI	Degree of Infeasibility
r_{in}	Number of process input resources	R_{in}	The set of process input resources
r_{out}	Number of process output resources	R_{out}	The set of process output resources
t_{in}^i	Number of input resources of task i	T_{in}^i	Set of input resources of task i
t_{out}^i	Number of output resources of task i	T_{out}^i	Set of output resources of task i
p	Number of task/process attributes	TA_i	Attribute values of task i
		PA	Process attribute values
c	Number of container attribute values	CA	Container attribute values

Table 6.1 Parameters for business process process optimization problem

The multi-objective problem formulation for BPO is as follows:

For a business process design with a set of n_d tasks and p process attributes:

Minimize/maximize $PA_i, i \in \{1, 2, \dots, p\}$

Subject to:

1. $DoI = 0$
2. $n \geq n_{max} \geq n_d \geq n_{min} \geq 0$
3. $n \geq n_c \geq 0$
4. $r \geq r_{in}, r_{out}, t_{in}^i, t_{out}^i > 0, i \in \{1, 2, \dots, n_d\}$
5. $p \geq 2$
6. $c = p + 1$

It is assumed that the process attributes serve as the optimization objective. A process attribute (PA_j) can be calculated as the aggregate of the corresponding task attributes for the n_d tasks in the process design according to the following equation.

$$PA_j = \sum_{i=1}^{n_d} TA_{ij} \text{ (Equation 6.1)}$$

In addition, there are more than one process attributes used as optimization objectives, hence it is considered as a multi-objective optimization problem. Furthermore, every process design is extracted by the n_c tasks of a container in the population. The aim of eBPO_F is to evolve the containers in the population in a way that makes them more promising for constructing a feasible optimized process design with their tasks. Therefore, it is assumed that the container attribute values and the DoI serve as the evolution objective. A container attribute value (CA_j) is calculated as the aggregate of the n_c tasks in the container with an equivalent equation of equation 6.1.

$$CA_j = \sum_{i=1}^{n_c} TA_{ij} \text{ (Equation 6.2)}$$

The problem formulation also involves 6 mandatory constraints. Constraint (1) ensures that only *feasible* business process designs are constructed. The Degree of Infeasibility (DoI) is an extra minimization objective of eBPO_F, calculated by PCA-II and measures to which degree a N_d subset of the N_c set forms a feasible business process design. A feasible design always has DoI equal to zero as the corresponding container does. Constraint (2) ensures that the library of tasks has more available tasks or at least those needed to compose a feasible design (n_d) and that both (n, n_d) are greater than zero. It also sets a lower (n_{min}) and an upper limit (n_{max}) to the number of tasks (n_d)

that can formulate a feasible design. Constraint (3) ensures that the library of tasks has more available tasks or at least those needed to fill a container (n_c) and that both (n, n_c) are greater than zero. Constraint (4) ensures that all resource-related parameters are greater than zero and the resources involved in a feasible design are available. Constraint (5) ensures that the problem is multi-objective or at least bi-objective. Finally, constraint (6) ensures that DoI is an extra objective for the container evolution.

6.2 Framework Overview

This section provides the functional overview of eBPO_F. The main components of the proposed framework are:

- i. the proposed business process representation technique
- ii. a series of Evolutionary Multi-objective Optimization Algorithms (EMOAs)

The proposed business process optimization framework applies a series of existing EMOAs to a business process design captured using the proposed representation. The outcome of the framework is a series of alternative optimized designs again in the form of the proposed business process representation. The challenge of the framework is to utilize the proposed representation technique and the capabilities of EMOAs efficiently, in order to generate alternative optimized designs.

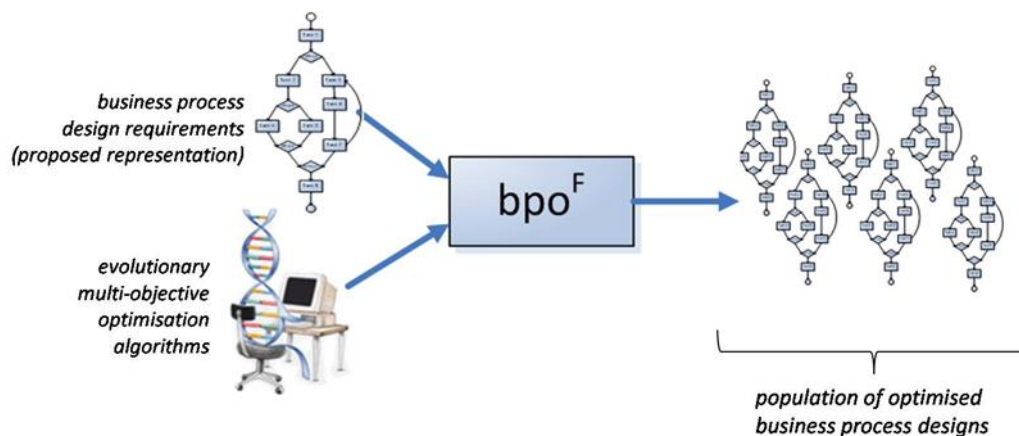
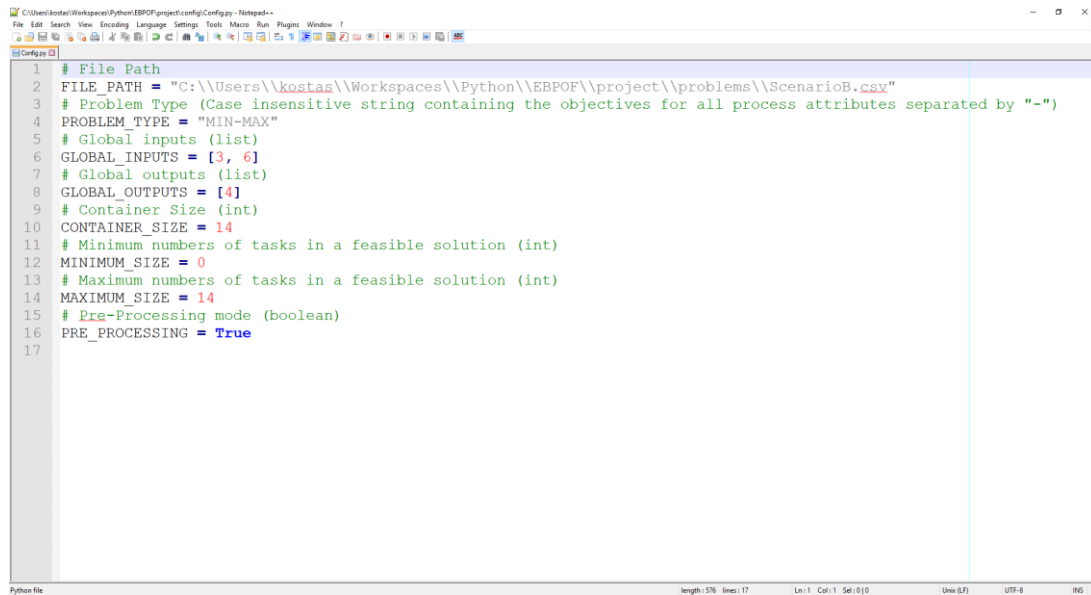


Figure 6.1 The main components of the proposed optimization framework by Vergidis (2008)

6.2.1 Main Operation, Inputs & Outputs

According to Vergidis (2008), the aim of such an optimization framework is to apply state-of-the-art EMOAs to given business process requirements, in order to generate a series of alternative optimized designs. Based on the aim, the main operation of the framework is the generation and optimization of business process designs. To achieve this, the user is responsible to provide a configuration file to the framework and this in turn is responsible to produce the alternative optimized designs. The configuration file of eBPO_F is shown in figure 6.2.



```

1 # File Path
2 FILE_PATH = "C:\\Users\\kostas\\Workspaces\\Python\\EBPOF\\project\\problems\\ScenarioB.csv"
3 # Problem Type (Case insensitive string containing the objectives for all process attributes separated by "-")
4 PROBLEM_TYPE = "MIN-MAX"
5 # Global inputs (list)
6 GLOBAL_INPUTS = [3, 6]
7 # Global outputs (list)
8 GLOBAL_OUTPUTS = [4]
9 # Container Size (int)
10 CONTAINER_SIZE = 14
11 # Minimum numbers of tasks in a feasible solution (int)
12 MINIMUM_SIZE = 0
13 # Maximum numbers of tasks in a feasible solution (int)
14 MAXIMUM_SIZE = 14
15 # Pre-Processing mode (boolean)
16 PRE_PROCESSING = True
17

```

Figure 6.2 Configuration file of eBPOF

The configuration parameters above form the following inputs for the framework:

1. The *process requirements* which consist of the process input resources (R_{in}) and the process output resources (R_{out}). All feasible designs must utilize some tasks from R_{in} and end up with all tasks of R_{out} .
2. The *container size* (n_c). The container size denotes the number of tasks in the candidate containers. Every solution in the population maintained by the used EMOA is perceived as a “container” of n_c tasks from which the framework attempts to compose feasible designs of n_d tasks through generations and thus, during the optimization process, the framework can generate designs with fewer tasks.
3. The *library of tasks* (N). This set contains all tasks that can potentially participate in a process design. Given the container size, a candidate container is formed with n_c tasks from the library.
4. The *minimum size* (n_{min}). The lower threshold for the number of tasks that a feasible design must have in order to be valid according to the user’s intention.
5. The *maximum size* (n_{max}). The upper threshold for the number of tasks that a feasible design must have in order to be valid according to the user’s intention.
6. The *problem type* specifies the optimization objective for every process attribute.
7. The *pre-processing mode* indicates whether a pre-processing operation will be performed on the library of tasks or not.

As it is shown in figure 6.2, there is a parameter called “*FILE_PATH*” which is assigned to a csv file. This file contains the specification of tasks from which the library of tasks will be created. An example of such a csv file is presented in figure 6.3 for “ScenarioC” problem.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	Inputs	Outputs	Attribute1	Attribute2																
2	2	3	209	106																
3	1	0	212	114																
4	2 1	3 0	227	102																
5	2	3	210	105																
6	2	3	212	109																
7	1	0	201	101																
8	2 1	3 0	207	110																
9	2	3	208	112																
10	5	2 1	205	103																
11	1	0	215	109																
12	2	3	228	108																
13	5	2 1	228	104																
14	2 1	0	213	115																
15	3 0	4	216	106																
16	5	2 1	219	109																
17	5	2 1	224	104																
18	5	2 1	225	113																
19	1	0	211	109																
20	5	2 1	211	103																
21	1	0	224	110																
22	2	3	204	112																
23	3	3	211	111																

Figure 6.3 Part of ScenarioC csv file

In this file, each record refers to a specific task. The first two columns contain the task input and output resources, respectively. Multiple resources are separated with the pipe symbol, “|”. The other columns refer to the task attributes; one column for every attribute. The task attributes are used for the calculation of the process and the container attribute values according to the equations 6.1 and 6.2, respectively.

```

1 from project.config.Config import *
2
3
4 # Algorithm name (string)
5 ALGORITHM = "NSGA2"
6 # Population Size (int)
7 POPULATION_SIZE = 250
8 # Probability for crossover operation (float)
9 CXPB = 0.8
10 # Probability for mutation operation (float)
11 MUTPB = 0.2
12 # Generation Size (int)
13 GENERATION_SIZE = 50
14
    
```

 Figure 6.4 Configuration file for NSGA₂

The proposed optimization framework applies a series of Evolutionary Multi-objective Optimization Algorithms (EMOAs) in order to generate optimized business process designs. The selected EMOAs in eBPO_F are: NSGA₂, SPEA₂ and DCD. All selected algorithms are state-of-the-art, and each has distinctive features that enhance the optimization process. These algorithms perform the *selection* operation and they will be discussed later in this section. Employing a range of EMOAs provides the opportunity to the user to compare their performance and their suitability

for the examined problem. For each of the selected EMOAs the user is responsible to provide a configuration file with a set of parameters that control the optimization operation. The corresponding configuration file for NSGA₂ is shown in figure 6.4 above.

The optimization operation is controlled by the following parameters:

1. The *population size* which denotes the number of the containers maintained, evolved and evaluated by the framework through generations.
2. The *generation size* which specifies the number of generations that the population will be evolved and evaluated.
3. The *crossover probability*. The probability of the crossover operation to be performed on a specific container.
4. The *mutation probability*. The probability of the mutation operation to be performed on a specific container.

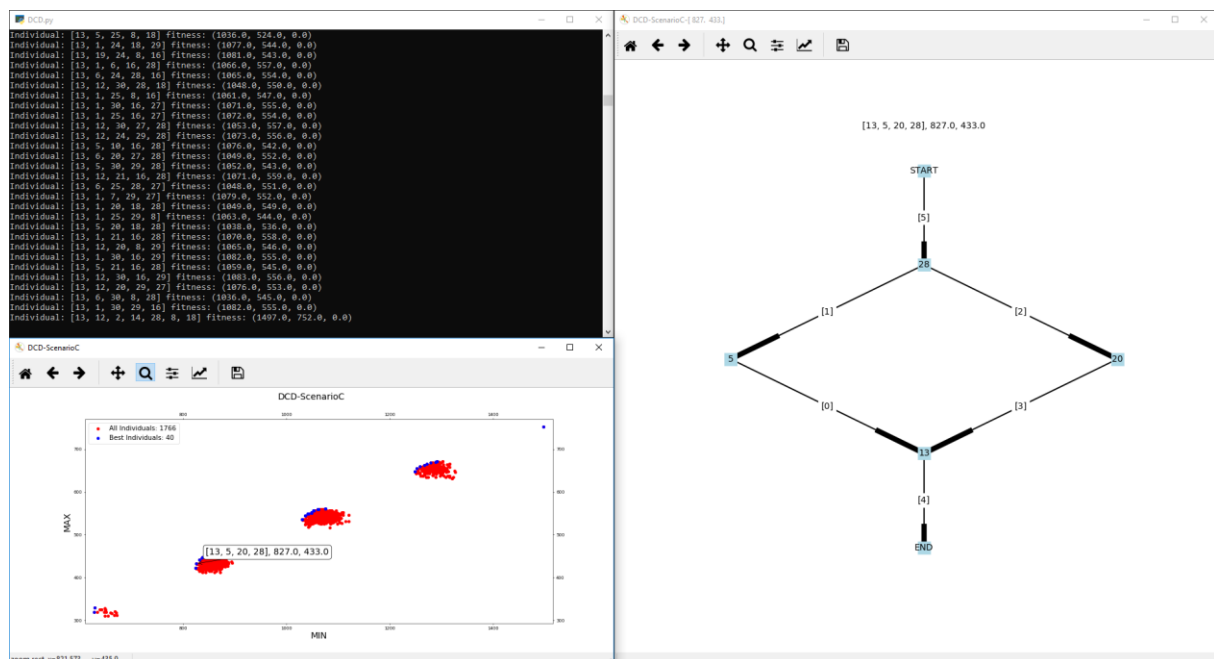


Figure 6.5 Outcomes of eBPO_F execution

The proposed optimization framework generates a series of optimized business process designs using the inputs in conjunction with one of the evolutionary algorithms. The whole operation is discussed step-by-step in the following sub-section. The outcome of the framework is a data structure called “Hall of Fame” which contains all feasible business process designs found during the execution of eBPO_F. From those designs, their pareto front is produced which comprises the optimized process designs found by eBPO_F.

For each design, Hall of Fame holds:

1. The *tasks in the design*, stored in the N_d set
2. The *process graph*, which is the diagrammatic representation of that design
3. The *Degree of Infeasibility (DoI)*, which is equal to zero for all feasible designs (for validation purposes)
4. The *process attribute values*, which are calculated based on the equation 6.1. These are the objective values which quantitatively show the performance of the design based on the type of the examined problem.
5. The *container attribute values*, which are calculated based on the equation 6.2. These constitute the evolution objectives of the containers which quantitatively express the ability of the containers to evolving towards a container with a more optimized feasible design.

Figure 6.5 shows the population of feasible designs after the framework execution for a business process optimization problem and the process graph of a non-dominated solution. Red dots represent the population of all feasible solutions found by the framework and the blue ones represent the non-dominated population of solutions which is the desideratum in BPO. Additionally, we can see that the *actual size* of the feasible solutions may differ even though the *containers* which those solutions occurred from, have the same *container size*.

The generation of business process diagrams is not an outcome explicitly included in the problem formulation. The sole outcomes requested by the problem formulation are the process attribute values which constitute the optimization objectives and the degree of infeasibility which denotes whether a process is feasible or not. All outcomes of the framework are the result of the *process composition algorithm*. Considering the problem formulation, PCA-II is triggered for each solution-container of the population through generations to compose a feasible process diagram. These aspects are clearer in the previous chapter where the new process composition algorithm, PCA-II, has been presented.

6.2.2 Main Steps of eBPO_F

The main steps and the structure of the proposed business process optimization framework are presented in the figure 6.6. The framework employs a generic optimization structure whose selection operation can be handled each time by a specific EMOA. It starts by setting-up the problem specification from the user inputs and the necessary data structures during executing. The next step is the optional pre-processing on the library of tasks. After pre-processing, the population of containers is created using the remaining valid tasks and evaluated. Finally, the optimization steps are executed for a predefined number of generations. Furthermore, each of the

optimization steps is adjusted to reflect the business process problem and ensure that the framework utilizes the process inputs and produces the process outputs. This sub-section describes the generic optimization process and the business process-oriented adjustments in each step while the next sub-section discusses the details of the framework operation for the selected EMOAs.

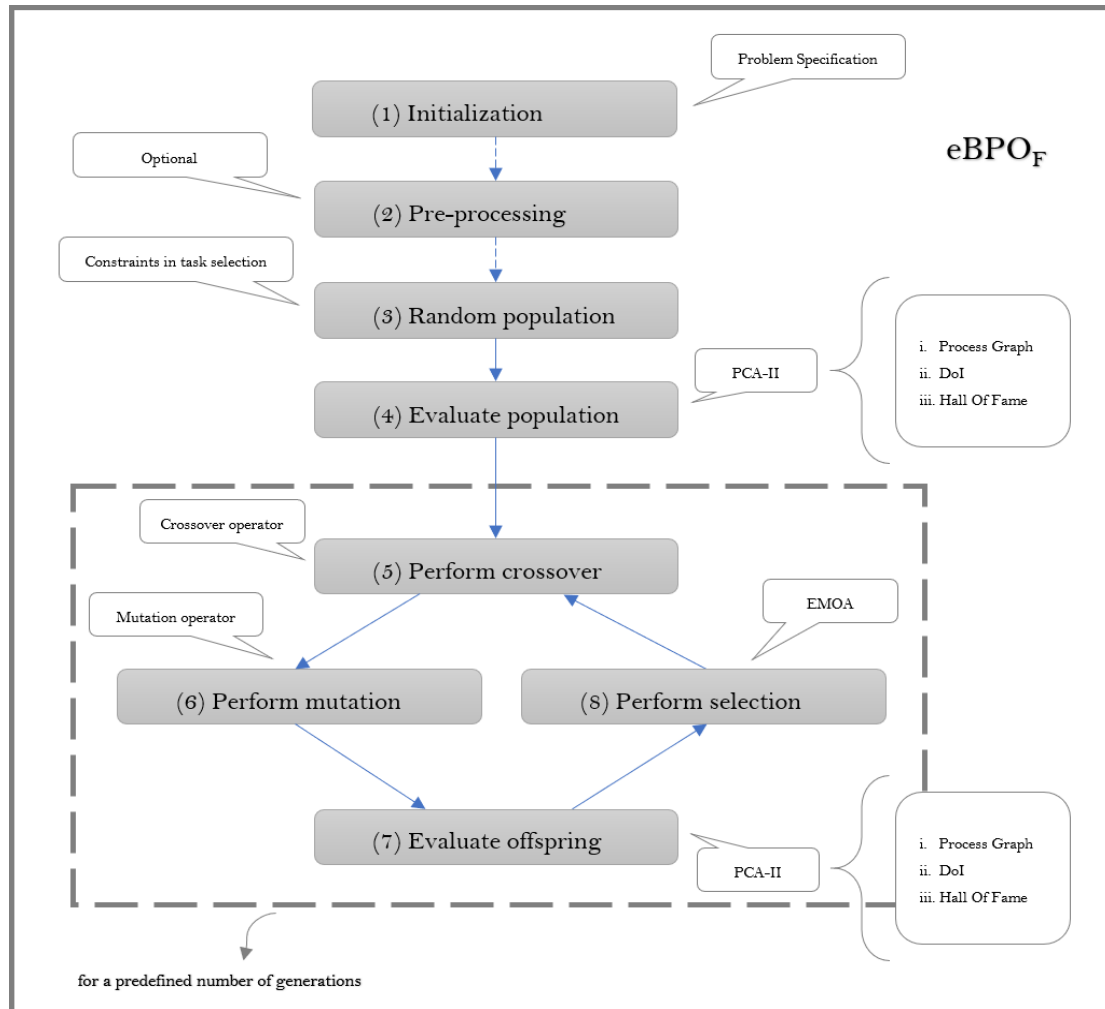


Figure 6.6 Main steps of eBPO_F

6.2.2.1 Initialization

The first step is to set-up the problem specification from the user inputs and prepare the necessary data structures for executing. All problem specification is stored in a dictionary named as “*problemSpecs*” to facilitate the flow of information within the framework. This dictionary has the records presented in figure 6.7.

```

1  problemSpecs['librarySize']: The number of available tasks
2  problemSpecs['validTasks']: Boolean array. True if the corresponding task in the library is valid after pre-processing
3  problemSpecs['numOfAttributes']: The number of objectives
4  problemSpecs['type']: The problem type
5  problemSpecs['inputs']: Dictionary of integer arrays with the input resources of each task
6  problemSpecs['outputs']: Dictionary of integer arrays with the output resources of each task
7  problemSpecs['attributes']: Dictionary of float arrays with the attribute values of each task
8  problemSpecs['resources']: The number of available resources
9  problemSpecs['resourcesSize']: Integer array holding the available resources
10 problemSpecs['resourcesDict']: Dictionary with original resources as keys and the corresponding resource indexes as values
11 problemSpecs['globalInputs']: Integer array holding the process input resources
12 problemSpecs['globalOutputs']: Integer array holding the process output resources
13 problemSpecs['minSize']: The minimum number of tasks that a feasible process design must have
14 problemSpecs['maxSize']: The maximum number of tasks that a feasible process design must have
15 problemSpecs['containersSize']: The number of tasks that the containers in the population must have
16 problemSpecs['populationSize']: The number of containers in the population
17 problemSpecs['cxbp'] = The probability of the crossover operator to be performed on a container
18 problemSpecs['mutpb'] = The probability of the mutation operator to be performed on a container
19 problemSpecs['generationSize'] = The number of generations that the population evolves
20 problemSpecs['preProcessing'] = True if pre-processing is going to be performed in the library of tasks
21 problemSpecs['scenario'] = The name of the examined problem
22 problemSpecs['algorithm'] = The name of the selected EMOA
23 problemSpecs['hallOfFameDict'] = Dictionary holding all feasible solutions found during executing along with their process graph
24 problemSpecs['uniqueSolutions'] = Set of the unique feasible solutions found during executing
    
```

Figure 6.7 The dictionary with the problem specification

6.2.2.2 Pre-processing

After the initialization step, there is the optional step of the pre-processing of the library of tasks and the user is responsible to choose whether this operation is going to be executed or not. The aim of this step is to search in the library of tasks for inconsistencies with the problem requirements and the objectives. If there are indeed inconsistent tasks, they will be made invalid for the optimization steps afterwards. In addition, this step examines if at least one feasible design can be composed by the remaining valid tasks in the library according to the problem requirements and signifies whether the optimization operation should be performed or not. The various aspects of the pre-processing operation have been discussed thoroughly in chapter 4.

6.2.2.3 Generate random populations

The first step of the optimization process is the generation of two random populations; the need for the second population has been discussed in the previous chapter where the PCA-II is presented. This step occurs only once in the optimization process and then these populations are evolved for a predefined number of generations. For the first population, PCA-II, follows the non-greedy composition approach where it attempts to construct a feasible process design with as less as possible tasks from the corresponding container. On the contrary, the second population is evaluated with the greedy approach where PCA-II attempts to construct a feasible process design as much as elaborate it can. The second population is the one where the “OR” patterns inside the process designs, are formed. The random populations consist of a fixed number of containers of n_c tasks. The number of the containers generated for each population, is equal to the *population size* that the selected algorithm is working with. Each of the container in the population contains n_c randomly selected tasks from the library of tasks, N . The only constraint in the random selection of the tasks is that *a task must appear only once in the same container*. This constraint avoids having duplicate tasks in a container of the initial populations, but as it will be discussed later, a container could have duplicate tasks through generations because of the genetic operators. However, a potential business process design cannot have duplicate tasks, and this is preserved by PCA-II in

the whole optimization operation since it cannot attach a new task from the container to the design if an identical is already attached to it.

6.2.2.4 Evaluate populations-offspring

After the random populations have been generated and before the populations for the next generation are selected, they should be evaluated. At this step, the process composition algorithm is executed for every container in the populations. PCA-II attempts to compose a feasible business process design with the tasks contained in the solution-container evaluated, based on the process requirements and the rules that such a feasible design shouldn't violate. These rules have been discussed in the previous chapter where PCA-II is presented. The outcome of the PCA-II is the diagrammatic version of the business process design composed from the container, the *actual tasks* in the composed design and the DoI of the design. The actual tasks in the design and the process design composed by these tasks will be used to update the *Hall of Fame* if the process design is feasible. Hall of Fame is a new feature coming with PCA-II and is one of the main differences between this process composition algorithm and its predecessor. Hall of Fame is the data structure that contains all feasible solutions found through generations and it has been further discussed in the previous chapter where PCA-II is presented. The array of the actual tasks is necessary because the feasible process design can occur from a subset of the n_c tasks of the solution-container evaluated, and only their attribute values are considered in the process attributes.

PCA-II ensures that there is one-to-one relationship between the solution evaluated and the feasible design that may occur to ensure the consistency in the optimization process. This means that for a **sequence** of n_c tasks the **same** feasible process design with n_d tasks, will be produced, if any, every time that PCA-II is executed. Again, this aspect gets clearer in the previous chapter where PCA-II is presented. At this stage of the optimization process, a process design has been created by PCA-II, its DoI has been measured and if it is feasible, DoI equals to zero, it will be stored in the Hall of Fame. The last part of this step is to update the attribute values of the container evaluated. This has to do with the selection operation performed by the selected EMOA at each generation and this time, the attribute values of all n_c tasks are considered in the container attribute values along with the DoI of the produced design. The **DoI** in eBPO_F serves as an **additional objective** apart from those coming with the problem examined, and the framework always tries to **minimize** it; a “feasible” container must have DoI equal to zero. This is another new feature of eBPO_F discussed in the previous chapter. After the random populations have been evaluated, steps 5–8 are repeated for a predefined number of generations.

6.2.2.5 Perform crossover

At this step, the *crossover* operation is performed on both populations. Crossover is a genetic operator used for exchanging information between two parents to generate new offspring. The crossover operator occurs directly in the N_c set of each container and each of the child containers contains tasks from both the parent containers. Figure 6.8 demonstrates the result of the crossover operator on two containers with $n_c = 6$ tasks. Initially, adjoining containers in the population are selected for crossover based on the *crossover probability*, defined separately by each of the EMOAs in the corresponding configuration file. The containers chosen for crossover, are split into pairs. For each pair, a unique crossover point is defined based on a random number between 1 and $n_c - 1$. Based on this crossover point, the parent containers exchange their tasks after that point, in order to form the child containers.

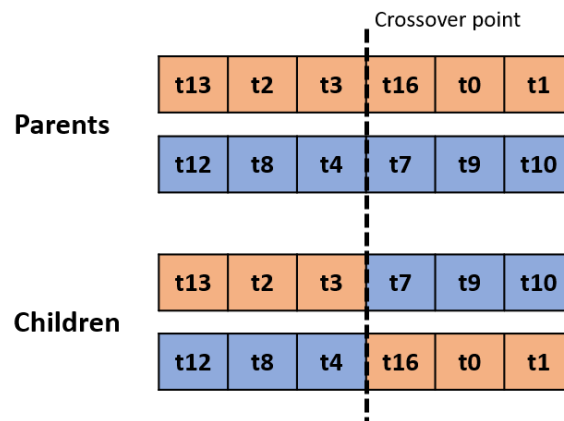


Figure 6.8 The crossover operator

After crossover, a child container could have duplicate tasks, however, the process design composed from that container, should not, and this is something preserved in the composition approach followed by PCA-II.

6.2.2.6 Perform mutation

After crossover, the *mutation* operator is performed on both populations. Mutation is a genetic operator used for altering information in a chosen solution-container. Similar to crossover, the mutation operator is applied to the N_c set of each container and each of the child containers has an altered order of tasks. Figure 6.9 shows the result of the mutation operator on a container with $N_c = 6$ tasks. Initially, each container in the population is selected for mutation based on the *mutation probability*, defined separately by each of the EMOAs in the corresponding configuration file. Then, each task in the container, is visited and is selected with a constant-defined probability of **0.5** to change position within the container, with another arbitrary task of the same container. Generally, the probability should be set low because if it's too high the optimization process will turn into a primitive random search. So, each task in the container has $(0.5 * \text{mutation probability})$ to chance its position within the container.

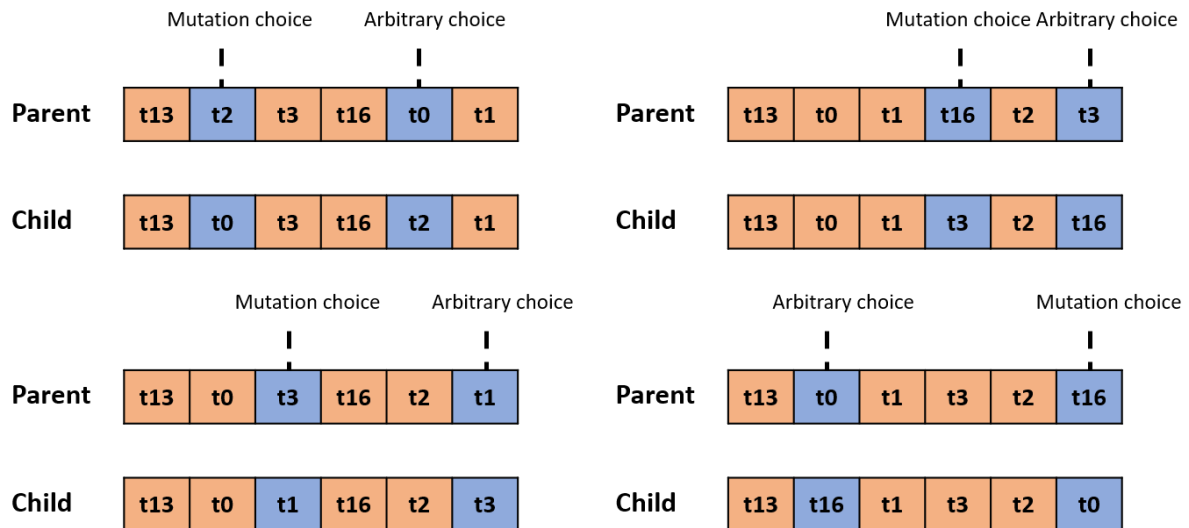


Figure 6.9 The mutation operator

6.2.2.7 Perform selection

The step of the optimization process which constitutes the beginning of a new generation is the *selection*. At this step, the next parent populations are formed by selecting the most promising containers for constructing a feasible optimized process design with their tasks, from the parent populations of the current generation and the child populations. The population size is a number defined separately by each of the EMOAs in the corresponding configuration file. The selection is performed based on the container attribute values, the problem type and the minimization of DoI which is an extra objective in eBPO_F for the evolution of the containers. Finally, the EMOAs are responsible for this step and the main difference among them, is the selection strategy that they follow. The choices for EMOAs in eBPO_F are presented in the next section.

6.2.3 EMOAs used in eBPO_F

In eBPO_F, the optimization algorithms employed are: NSGA₂, SPEA₂ and DCD. NSGA₂ and SPEA₂ are the most prominent EMOAs used when comparing a newly designed EMOA, Coello Coello (2005). These algorithms are responsible for performing the selection of the most promising containers to be the next parent population. The new optimization framework is structured in a way that it can operate with any of the three employed EMOAs. This section describes how each of the EMOAs manages the framework differently and what the different impact is. Each EMOA is different in two main areas:

1. The selection operator process, i.e. fitness assignment and constraint handling
2. The type of parameters that it uses, e.g. population size, number of generations, crossover and mutation probability

6.2.3.1 Non-dominated Sorting Algorithm 2 (NSGA₂)

NSGA₂ is one of the three EMOAs incorporated in eBPO_F. It is considered as a high-performing multi-objective optimization algorithm and was developed by Deb *et al.* (2001) as an answer to the criticisms of the original NSGA. It is an elitist algorithm that uses a parent and a child population in each generation in order to maintain “good” solutions and has provided satisfactory results in real world applications. The diversity among non-dominated solutions is introduced in NSGA₂ by using the crowded comparison operator for selecting the parent solutions for the next generation. The crowded comparison operator guides the selection operation towards a uniformly spread-out Pareto front. However, NSGA₂ is known for not performing well in problems with multiple local fronts. The fitness assignment strategy of NSGA₂ ceases to produce the driving force towards the global front once most of the solutions of the population share the same non-domination level. Furthermore, the use of elitism enhances this behavior of NSGA₂ which tends to get trapped in local fronts (pre-mature convergence). eBPO_F optimizes containers for constructing business process designs of different sizes thus creating multiple local fronts. Utilizing, NSGA₂ will examine its capability of discovering and optimizing solutions of variables sizes in terms of business process designs. The main parameters of NSGA₂ are the population size, the number of generations along with the crossover and mutation probabilities.

6.2.3.2 Strength Pareto Evolutionary Algorithm 2 (SPEA₂)

SPEA₂ is also incorporated in eBPO_F and is another elitist evolutionary algorithm. It was developed as the improved version of SPEA by the same group of researchers, Zitzler *et al.* (2001). SPEA₂ has been popular in the evolutionary multi-objective optimization society and has been used in a variety of optimization problems. It works by maintaining an external population at every generation storing all non-dominated solutions discovered so far beginning from the initial population that participates in all genetic operations. SPEA₂ uses a selection strategy in which a “strength” is associated with each member of the archive. The “strength” of a solution is based on the number of solutions in the internal population which it dominates. Selection is biased towards minimizing the strength of the solution thus preferring the exploration of less populated regions of the search space. Because of this strength selection mechanism, it is expected that SPEA₂ will demonstrate flexibility in converging to optimal solutions across the search space. The main parameters of SPEA₂ in eBPO_F are the (external) population size, the number of generations and the crossover and mutation probabilities.

6.2.3.3 Tournament selection based on Dominance and Crowding Distance (DCD)

The third evolutionary algorithm employed in eBPO_F is DCD. This algorithm performs a tournament selection based on the dominance between two solutions and if these solutions interdominate, the selection is made based on their crowding distance. Initially, two random

samples of the population are created. Then, the dominant one of each two consecutive containers for each population sample, is selected. If there is no dominant task, it is selected the one with the biggest crowding distance, the average distance of its two neighboring solutions. Finally, if there is still no better solution, one of the two is arbitrarily selected. The tournament selection has several benefits over alternative selection methods for genetic algorithms (e.g. fitness proportionate selection and reward-based selection): it is efficient to code, works on parallel architectures and allows the selection pressure to be easily adjusted, Miller and Goldberg (1995). Selection pressure is a probabilistic measure of a solution's likelihood of participation in the tournament based on the participant selection pool size, is easily adjusted by changing the tournament size. If the tournament size is larger, weak individuals have a smaller chance to be selected, because, if a weak individual is selected to be in a tournament, there is a higher probability that a stronger individual is also in that tournament. Tournament selection has also been shown to be independent of the scaling of the genetic algorithm objective function in some classifier systems, Goldberg and Deb (1991). The main parameters of DCD in are the population size, the number of generations and the crossover and mutation probabilities.

6.2.4 Solution representation

As discussed in the previous chapter, eBPO_F evolves two populations of containers through generations to be more promising for constructing a feasible optimized business process design with their tasks. According to Vergidis (2008), there are three optimization challenges within the framework for the solutions:

1. A solution should meet the different requirements of each optimization stage (solution representation)
2. A solution should be able to accommodate designs of different sizes (solution size)
3. A solution should not restrain a design in terms of process patterns (design flexibility)

eBPO_F uses the containers to address the challenges above and ensures the consistency of the them during the process. In addition, the one-to-one relationship between the container and the produced process design is ensured by PCA-II. Specifically, this algorithm follows two different approaches for composing a feasible process design with the tasks in the container based on the population that the examined container belongs to. In both approaches, the tasks in a container are always visited from left to right in PCA-II, so the order of the tasks in the container is something that matters preserves the relationship mentioned above. The path of the containers across the different stages of eBPO_F are shown in figure 6.10.

At the beginning, the random populations are created with containers of N_c tasks. Then, for every container in the populations, PCA-II is triggered to compose a feasible process design with their

tasks. At this stage, a set of N_d tasks is created along with the corresponding process design. In addition, the DoI of the produced design is calculated and the container attribute values are updated. This design refers to the most elaborated one found during PCA-II execution. Next the crossover and mutation operators occur in the N_c tasks of the parent containers and afterwards, PCA-II is triggered for the containers in the offspring population. Finally, the selection operator is performed to the former parent population and the offspring population to select the parent population for the next generation according to the container attribute values.

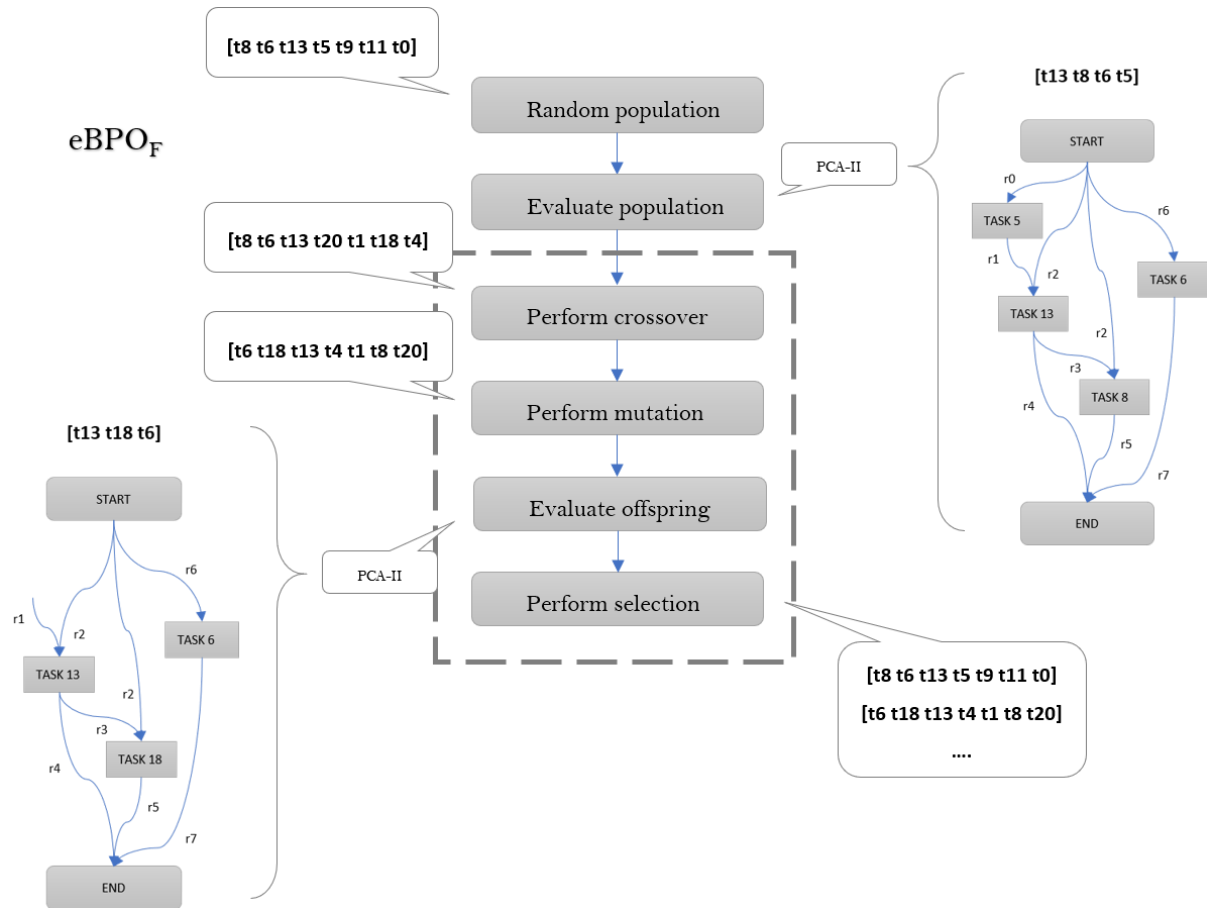


Figure 6.10 The container within eBPO_F

6.3 Framework Implementation

The new framework has been developed using the Python programming language. Python was selected because it is one of the most suitable programming languages for numerical computing and quick prototyping. It is an interpreted, high-level, general-purpose, object-oriented programming language which emphasizes on code readability and supports functional and imperative programming. In addition, it provides many well-maintained libraries for scientific computing, visualization, evolutionary algorithms and graphs which constitute the foundations of the new framework. eBPO_F was programmed using the four open-source Python libraries listed below and their usage within the framework is shown in figure 6.11.

1. NumPy, which stands for Numerical Python, is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. Some of its benefits are: the contiguous allocation in memory, the vectorized operations, the boolean selection and the sliceability
2. Matplotlib is a plotting library for the Python programming language which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms
3. DEAP, which stands for Distributed Evolutionary Algorithms in Python, is an evolutionary computation framework for rapid prototyping and testing of ideas. It seeks to make algorithms explicit and data structures transparent. It incorporates the data structures and tools required to implement most common evolutionary computation techniques such as genetic algorithms, genetic programming, evolution strategies, particle swarm optimization, differential evolution, traffic flow and estimation of distribution algorithms.
4. NetworkX is a Python package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks. Some of its features are: the data structures for graphs, digraphs, and multigraphs, many standard graph algorithms, drawing of 2D and 3D networks, well tested with over 90% code coverage, fast prototyping, easy to teach and multi-platform

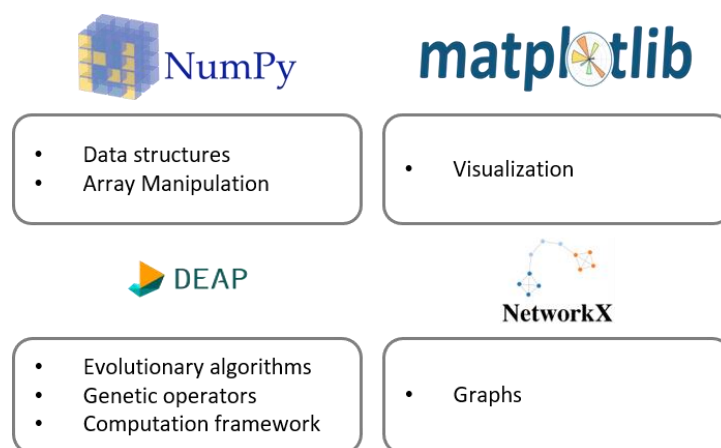


Figure 6.11 The libraries used in eBPO_F

Figure 6.12 shows the packages developed by the author for eBPO_F. The first package “config” stores the configuration files of the framework and the employed EMOAs. The packages “dsd”, “nsga2” and “spea2” as their name implies, define the framework operation under each of the EMOAs. The “init” package is responsible for the initialization phase and the “visualization” one for the interactive plots after execution. The “tools” package defines some auxiliary functions and the “evaluation” one performs the evaluation of the containers in both populations. Finally, the

packages “preprocess” and “pca2” contain the main contribution of the new framework in the domain of business process optimization, the pre-processing of the library of tasks and the new process composition algorithm respectively.

```

1  import ...
2
3
4
5
6
7  def attachTasks(attachedTasks, blackSet, globalInputs, Graph, container, newAttachedTasks, parentTask, resource,
8      taskOutputs, unusedTasks, greedy=False):
9
10     # Initialize inputSatisfied equal to False
11     inputSatisfied = False
12
13     ##### Try to find the resource from global inputs #####
14
15     # If the resource is a global input resource
16     if resource in globalInputs:
17
18         # Add an edge from start task to current parent task identified by the resource value
19         Graph.add_edge('START', parentTask, key=resource)
20
21         # If the resource has been satisfied return inputSatisfied which is equal to True
22         inputSatisfied = True
23         return inputSatisfied
24
25     #####
26
27     ##### Try to find the resource from tasks already inserted in the graph #####
28
29     # tmpAttachedTask = -1

```

Figure 6.12 Screenshot of the Python programming environment

6.4 Summary

This chapter introduced eBPO_F, the extended Business Process Optimization framework. This framework constitutes the new revised and improved version of BPO_F, the business process optimization framework introduced by Vergidis (2008). The proposed optimization framework has also been implemented in Python as a software tool and employs three different EMOAs to provide the opportunity for comparing their performance and their suitability for the examined problem. This chapter presented the user inputs and the expected outputs of the new tool, the steps of the optimization procedure and characteristics of the employed EMOAs. The next chapter presents the validation testing of the new optimization framework by examining the three real-life business process scenarios that have also been used for the validation of BPO_F.

CHAPTER 7

Testing & Results

This chapter presents the validation testing of the proposed optimization framework. The validation is made with three different real-life scenarios which have been developed by Vergidis (2008) for the validation testing of BPO_F. The chapter starts by presenting the specification of each scenario examined. Next, the results of the pre-processing operation on the library of each scenario are presented and a short discussion about them is provided. The chapter concludes with the end-to-end testing of eBPO_F for each scenario and the evaluation of the execution results.

7.1 Scenarios

This section introduces the real-life scenarios which the new framework is tested with. Vergidis (2008) has developed three scenarios for the validation testing of BPO_F by capturing the context of three different business processes of the service industry and creating the corresponding libraries of tasks by gathering relevant sub-services. The purpose of using real-life scenarios according to Vergidis (2008), is to validate the capability of the framework in capturing, composing and optimizing designs of business processes that are current practice in real-life situations. The aim of this thesis is to make the new framework capable of fulfilling more real-life constraints during the design composition and handling more complex problems. In addition, the motivation for the development of PCA-II was the unexpected results from the pre-processing of the library of tasks of those scenarios which correspond to real-life processes. Therefore, the validation of eBPO_F is made with the same real-life scenarios used by Vergidis (2008). These scenarios were selected by Vergidis based on the business process automation classification, aiming for showing the versatility and the capability of the framework to automate and optimize business processes of each level. Thus, each scenario belongs to a different classification in terms of automation:

1. Scenario A – Online order placement is an *automated* business process
2. Scenario B – Sales forecasting is a *semi-automated* business process
3. Scenario C – Fraud investigation is a *manual* process

All scenarios are considered as bi-objective optimization problems and their problem type involves the minimization of the first objective and the maximization of the second one. According to Vergidis, the first objective is the *Service Delivery Price*, (SDP), which specifies the amount of money that the service customer must pay to use the service. The second objective is the *Service Fulfilment Target*, (SFT), which specifies the service provider’s promise of effective and seamless delivery of the defined benefits to any authorized service customer requesting the service within the defined service times. It is expressed as the promised maximum number of successful individual service deliveries considering the total number of individual service deliveries.

7.1.1 Scenario A: Online order placement

The first scenario refers to the business process of placing an order in an online store, Havey (2005). It was an automated process already implemented by an end-to-end integrated application and the aim of this problem was to show the optimization potential of the framework for an automated business process. Figure 7.1 shows the business process design draft of Scenario A.

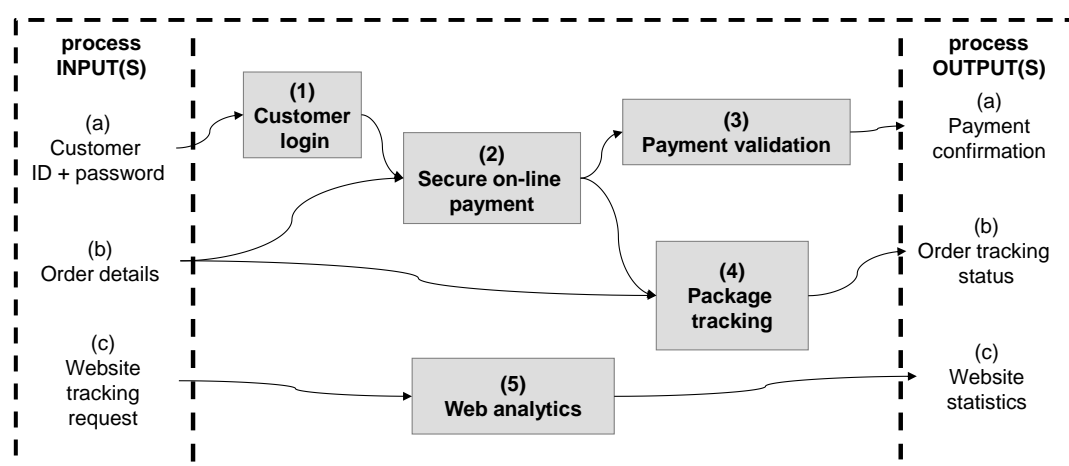


Figure 7.1 Business process design draft for online order placement by Vergidis (2008)

The design involves the main steps, how they are connected, and the identification of the process input and output resources. The scenario starts with three available process input resources: (a) Customer ID & password, (b) Order details and (c) Website tracking request. The main steps of the process are five. Initially, the customer credentials are necessary to access the online store, (Customer login), along with the order details to place the order and pay for it, (Secure online payment). Paying for the order invokes the payment validation, (Payment validation) and the monitoring of the order process, (Package tracking), which also needs the order details. In addition, the web analytics tracks the customer’s actions in the website, (Web analytics). The three expected output resources of the process are: (a) Payment confirmation which returns the payment status of the order, (b) Order tracking status which returns the order status in terms of delivery to the customer and (c) Website statistics which record the customer’s behavior in the website and influence the store’s marketing strategy in terms of customer’s individual needs.

7.1.2 Scenario B: Sales forecasting

The second scenario describes the business process of sales forecasting, Grigori *et al.* (2004). It was considered as a semi-automated process as it involved the interaction of some applications but was not streamlined and still required the human factor for generating and visualizing the requested forecasts. According to Vergidis, the aim of this problem was to fully automate the process by selecting and implementing relevant web services and propose a set of optimized designs that fulfil the process requirements having optimal attribute values.

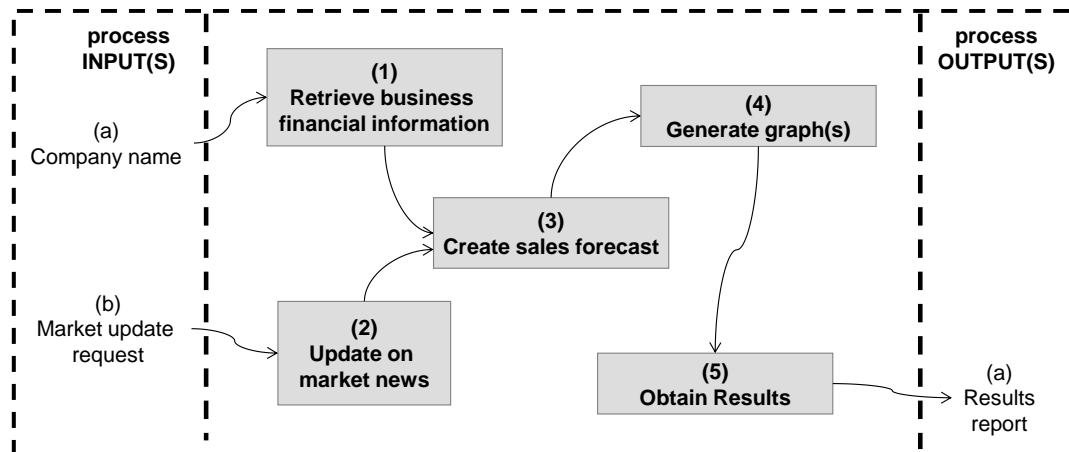


Figure 7.2 Business process design draft for sales forecasting by Vergidis (2008)

Figure 7.2 shows the business process design draft of Scenario B. The available process input resources for this scenario are: (a) Company name and (b) Market update request. The process design comprises five steps. Initially, the relevant business financial information is extracted, (Retrieve business financial information), and the market levels are updated to be considered for the new forecast, (Update on market news). Next, the outcomes of those two steps are fed to a Monte-Carlo simulation to generate the new forecast, (Create sales forecast). Then, a graph is constructed for visualizing the new forecast, (Generate graph(s)), and is communicated to the person requesting it (Obtain results). The only expected output resource of this process is: (a) Results report, which conveys the forecast results.

7.1.3 Scenario C: Fraud investigation

The last scenario concerns the business process of fraud investigation, Havey (2005), which occurs when there is a suspicion of customer identity fraud and consequent loss by misusing company goods and services. This process was considered as manual process because there was no standard procedure followed when investigating since there was no complete software application that could track, identify or prevent fraud. As a result, fraud investigation involved manual investigation of the data maintained by the company. According to Vergidis, the aim of this problem was to standardize the process, make it more reliable, automate it and optimize it. The process design draft for Scenario C is shown in figure 7.3.

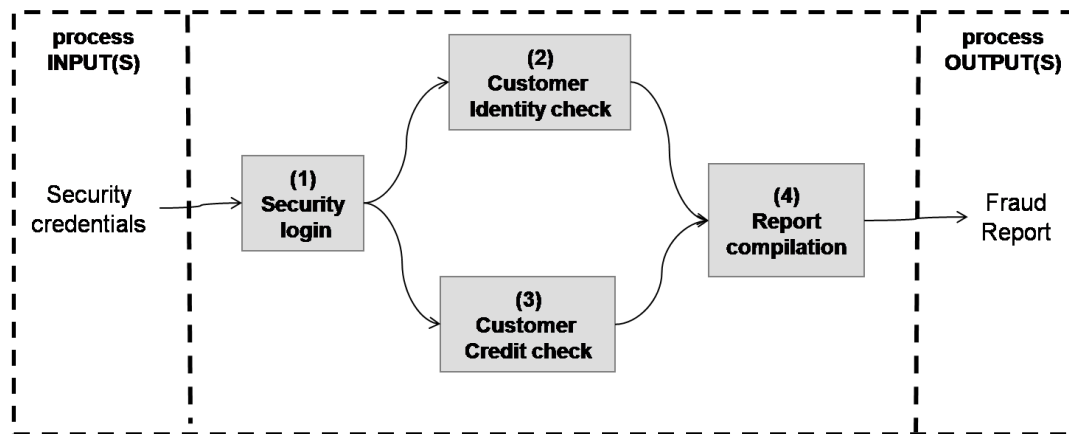


Figure 7.3 Business process design draft for fraud investigation by Vergidis (2008)

The only requested process input resource is Security credentials. Initially, the user must provide the credentials for accessing the data, (Security login). Next, two parallel checks take place for the customer's identity, (Customer Identity Check), and for the customer's credit card, (Customer Credit Card). Finally, the outcomes of those checks merge into a report, (Report compilation). This report constitutes the expected output resource of this scenario and based on this report, the company can take further actions.

7.2 Validation of pre-processing

This section showcases the effect of pre-processing on the library of tasks for the three scenarios examined. For every scenario, the remaining library of tasks is presented and a short discussion along with some execution examples are provided for the justification of the results.

7.2.1 Pre-processing of Scenario A

Figure 7.4 demonstrates the library of tasks and the resources involved in online order placement process. The column of valid tasks has been added by the author of this thesis to indicate the valid tasks after pre-processing. Initially, the library comprised 29 tasks and after pre-processing there are only 16 left, achieving a decrease of the library size of 45%.

For this scenario, all invalid tasks are the result of "Check Categories" sub-process (chapter4) and are dominated by others. For example, the tasks 1, 8, 9 and 27 belong to the same category. However, the tasks 1, 9 and 27 are dominated by task 8 considering the minimization of the first attribute and the maximization of the second one, hence they became invalid for optimization. In this scenario, it is obvious that every feasible process design found by eBPO_F after pre-processing, will have the best task for "Package tracking" since it is the only remaining task that returns the "Order tracking status", which is one of the process output resources. Hence, the performance of the EMOAs in eBPO_F must be benefitted in this case.

No.	Task Name	Input(s)	Output(s)	SDP	SFT	Valid
0	Achworks Soap (T\$\$ - Rico Pamplona)	1, 2	3	208	113	✓
1	BAX Global Tracking Service	2, 3	5	219	109	
2	CDYNE Death Index	1, 3	4	229	115	✓
3	Credit Card Processor	1, 2	3,4	202	109	✓
4	D&B Business Credit Quick Check	1, 3	4	203	108	✓
5	Drupal authentication	0	1	200	103	✓
6	ecommsStats Web Analytics	6	7	218	112	
7	Entrust login	0	1	206	103	
8	FedEx Tracker	2, 3	5	211	109	✓
9	FedEx / UPS Package Tracking	2, 3	5	224	103	
10	FraudLabs Credit Card Fraud Detection	1, 3	4	220	113	✓
11	Google Analytics	6	7	218	107	
12	Google Checkout	1, 2	3	206	105	✓
13	GUID Generator	0	1	203	110	✓
14	Internet Payment Systems	1, 2	3	226	105	
15	LID login	0	1	222	114	✓
16	OpenID login	0	1	228	100	
17	Paypal online payment	1, 2	3	215	102	
18	Real Time Check Verification (T\$\$ - Rico Pamplona)	3	4	229	108	✓
19	Rich Payments NET	2	3, 4	208	105	✓
20	SAINTlogin users validation	0	1	219	105	
21	Servicetrack	6	7	212	113	✓
22	SmartPayments Payment	2	3	214	105	✓
23	Smartpayments CardValidator	3	4	206	107	✓
24	Strikelron Global Address Verification	1, 3	4	201	105	✓
25	SXIP login	0	1	203	105	
26	Typekey authentication service	0	1	224	114	
27	UPS Tracking	2, 3	5	225	109	
28	VeriSign Payment	1, 2	3	230	103	

No.	Resource name	Process I/O
0	Customer account credentials	I
1	Customer account details	
2	Order details	I
3	Payment details	
4	Payment confirmation	O
5	Order tracking status	O
6	Website tracking request	I
7	Website statistics	O

Figure 7.4 Library of tasks and resources for online order placement by Vergidis (2008)

7.2.2 Pre-processing of Scenario B

The library of tasks and the resources involved in sales forecasting process are shown in figure 7.5. The column of valid tasks has been added by the author of this thesis to indicate the valid tasks after pre-processing. Initially the library had 20 tasks, but after pre-processing only 14 tasks are left. For this scenario, the decrease of the library size, is about 30%.

No.	Task Name	Input(s)	Output(s)	SDP	SFT	Valid
0	D&B Business Verification	3,1	0,5	206	103	
1	Fax.com	8,2	4	220	103	✓
2	Gale Group Business Information	3,0	5	223	106	✓
3	Gale Group Business Intelligence	3,1	0,5	229	113	
4	GraphMagic's Graph & Chart Web Service API	5,8	2	203	107	✓
5	interfax.net	8,2	4	222	113	✓
6	Lokad Business time-series forecasting and analysis	5,7	8	228	110	✓
7	Midnight Trader Financial News	6,3	7	217	101	✓
8	Strikelron Company Search	3	3,0	230	114	✓
9	Strikelron Get Business Prospect	3,1	0,5	205	110	
10	Strikelron Lookup Business	3	3,0	201	110	✓
11	Wall Street Horizon Real-Time Company Earnings	3,1	5	210	105	
12	Xignite Get Balance Sheet	3,1	5	216	112	
13	Xignite Get Chart Url	8	2	228	110	✓
14	Xignite Get Chart Url Preset	8	2	228	101	
15	Xignite Get Growth Probability	5,7	8	215	109	✓
16	Xignite Get Market News Headlines	6	7	221	114	✓
17	Xignite Get Market Summary	6	7	203	112	✓
18	Xignite Get Topic Chart	3,5,7	8	218	112	✓
19	Xignite Get Topic Data	3,5,7	8,2	222	109	✓

No.	Resource name	Process I/O
0	Business details	
1	Business query	
2	Chart / graph	
3	Company name	I
4	Fax (on-line)	O
5	Financial data	
6	Market update request	I
7	Recent market trends	
8	Time-series forecast	

Figure 7.5 Library of tasks and resources for sales forecasting by Vergidis (2008)

For this scenario, the invalid tasks are mainly the result of the “Check Task Inputs” sub-process (chapter 4). There is no task providing the resource 1 which is necessary for the execution of the tasks 0, 3, 9, 11 and 12, therefore they became invalid. The task 14 is dominated by task 13 as both tasks belong to the same category, and it is the result of the “Check Categories” sub-process as in the previous scenario.

7.2.3 Pre-processing of Scenario C

Figure 7.6 presents the library of tasks and the resources involved in fraud investigation process. The column of valid tasks has been added by the author of this thesis to indicate the valid tasks after pre-processing. The pre-processing of this scenario has managed to decrease the library size about 68% since 10 tasks are still valid out of 31. For this scenario, all invalid tasks are the result of the “Check Categories” sub-process (chapter 4). For example, the tasks which have resource 2 as task input resource and resource 3 as output, they are dominated by the tasks 22 and 30.

No.	Task Name	Input(s)	Output(s)	SDP	SFT	Valid
0	Address Doctor Global Address Verification	2	3	209	106	
1	cbarron bankValidate	1	0	212	114	✓
2	CDYNE Death Index	2, 1	3, 0	227	102	
3	CDYNE Email Verifier	2	3	210	105	
4	CDYNE Phone Verifier	2	3	212	109	
5	D&B Business Credit Quick Check	1	0	201	101	✓
6	D&B Business Verification	2, 1	3, 0	207	110	✓
7	Dimple Email Address Validator	2	3	208	112	
8	Drupal authentication	5	2, 1	205	103	✓
9	Dun & Bradstreet Business Credit Quick Check	1	0	215	109	
10	Dun & Bradstreet Business Verification	2	3	228	108	
11	Entrust login	5	2, 1	228	104	
12	FraudLabs Credit Card Fraud Detection	2, 1	0	213	115	✓
13	Google Docs	3, 0	4	216	106	✓
14	GUID Generator	5	2, 1	219	109	
15	LID login	5	2, 1	224	104	
16	OpenID login	5	2, 1	225	113	
17	Real Time Check Verification (T\$\$ - Rico Pamplona)	1	0	211	109	✓
18	SAINTlogin users validation	5	2, 1	211	103	
19	Smartpayments CardValidator	1	0	224	110	
20	Strikelron 24-hour Accurate Residential Lookup	2	3	204	112	
21	Strikelron 24-hour Accurate Reverse Phone Lookup	2	3	211	111	
22	Strikelron Email Verification	2	3	215	113	✓
23	Strikelron Gender Determination	2	3	215	102	
24	Strikelron Global Address Verification	2	3	211	111	
25	Strikelron Reverse Phone Residential Intel	2	3	203	111	
26	Strikelron Reverse Residential Lookup	2	3	222	107	
27	SXIP login	5	2, 1	216	110	
28	Typekey authentication service	5	2, 1	206	114	✓
29	Web Services Security Monitor	5	2, 1	227	110	
30	webba E-Mail validator	2	3	202	112	✓

No.	Resource name	Process I/O
0	Credit assessment	
1	Customer Credit details	
2	Customer ID details	
3	ID verification outcome	
4	Risk Assessment Report	O
5	Security login credentials	I

Figure 7.6 Library of tasks and resources for fraud investigation by Vergidis (2008)

7.3 Validation of eBPO_F

After pre-processing, the optimization operation is executed for each scenario and for all employed EMOAs. This section presents the results of eBPO_F for the examined problems along with some graphs which serve as examples of the produced optimized process designs. Finally, a short discussion will be provided for each scenario for the justification of the results.

In order to have also a view of the performance of eBPO_F in dealing with real-life problems, the majority of the feasible solutions of each scenario should be found to shape the search space of those problems. For this reason, a *brute-force* approach has been followed for each scenario where every possible combination of n_c valid tasks after pre-processing, is evaluated by PCA-II. In order for the valid tasks only to be examined by PCA-II, the brute-force approach has been executed in 4 cycles for each scenario where each cycle corresponds to one of the four possible problem types of a bi-objective optimization problem, (MIN-MIN, MIN-MAX, MAX-MIN and MAX-MAX).

Each of the employed EMOAs has the same configuration for each scenario examined and this configuration is also partially considered by the brute-force approach to find the search space of the scenarios. The configuration for the validation testing of the new framework, is shown in figure 7.7. For each scenario, the new framework under each optimization algorithm, is executed for 10 independent runs and the results of a typical run are shown in corresponding diagrams.

Parameter	Value	Parameter	NSGA2	SPEA2	DCD
Container size	10	Population size	100	100	100
Minimum size	0	Generation size	20	20	20
Maximum size	10	Crossover probability	0.8	0.8	0.8
Problem type	MIN-MAX	Mutation probability	0.2	0.2	0.2

Figure 7.7 Configuration of eBPO_F for validation testing

7.3.1 Results for Scenario A

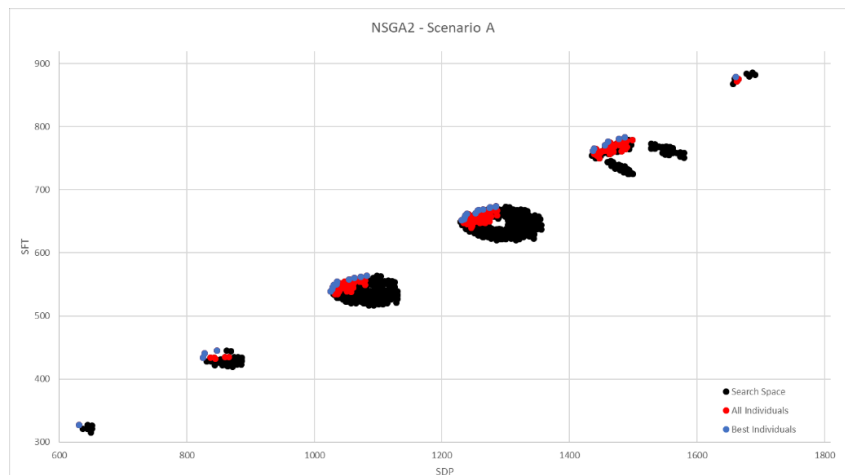


Figure 7.8 Scenario A under NSGA2

Figures 7.8, 7.9 and 7.10 display the optimization results of eBPO_F for the online order placement process for each EMOA. The black dots correspond to the search space, the red ones correspond to all feasible solutions found during executing and the blue ones refer to the Pareto front in terms of optimum solutions.

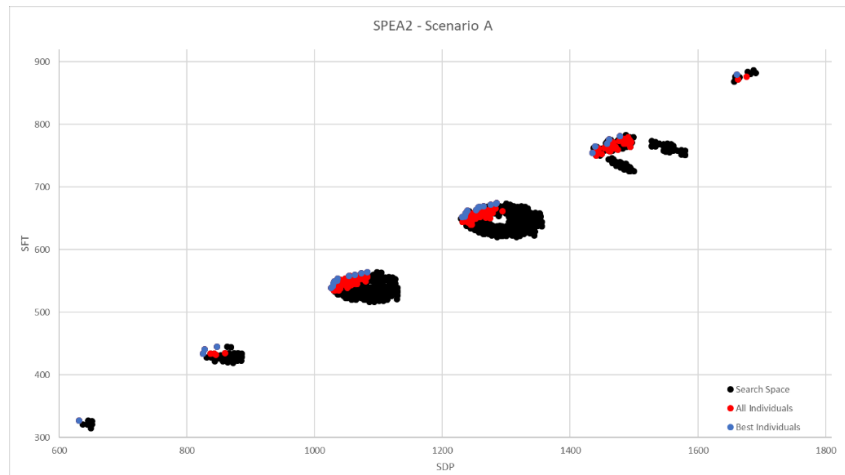


Figure 7.9 Scenario A under SPEA2

The optimization criteria are the minimization of the Service Delivery Price and the Service Fulfilment Target. Furthermore, the islands of solutions differ in the number of the tasks that their solutions have. As it is obvious from the results, the solutions found by the new framework, converge to and are spread out uniformly all over the Pareto-optimal front. In addition, the “hole” in the fourth island and the “gap” in the fifth island reveal the benefits of pre-processing on the search space of the examined problem.

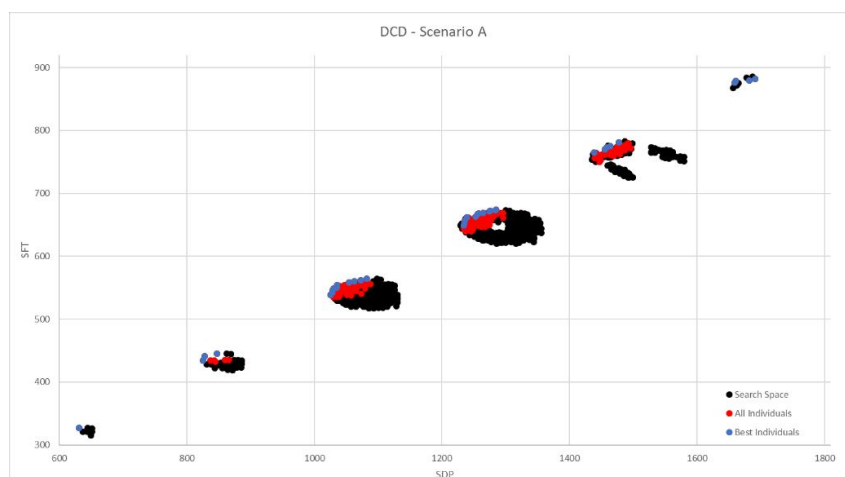


Figure 7.10 Scenario A under DCD

Figure 7.11 shows two optimized business process designs as they have been extracted by eBPO_F. These designs belong to different islands based on their solution size. The light-blue squares refer to the nodes and the numbers in the brackets depict the related resources. The left design of figure 7.11 slightly differs from the draft of figure 7.1 since a web service, tasks 3, can cover the steps 1

and ϱ simultaneously. Apparently, this feature of eBPO_F is very important for business process optimization since the framework can provide an alternative design with fewer steps, thus a design with lower cost and sufficient service level.

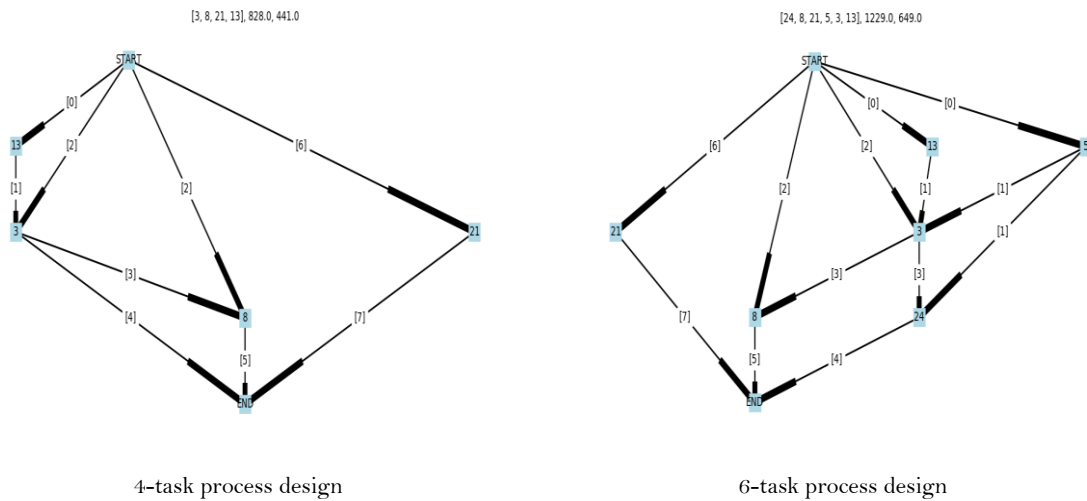


Figure 7.11 Optimized business process designs for Scenario A

On the other hand, the right design of figure 7.11, refers to a more elaborated design than the draft of figure 7.1. It provides two alternative suppliers for the customer account credentials needed by the secure online payment process making the whole process more reliable and thus achieving bigger SFT compared to the left design. However, the reliability has its cost and the right design has also bigger SDP compared to the left one.

Furthermore, figure 7.12 show the process design of the only feasible 3-task solution found by the framework, which is also the most optimal one according to the search space of Scenario A. In this process design, the process input resource 0, Credit assessment, is not exploited at all, but all process output resources can be produced. This is another new feature of eBFO_F where a feasible design can utilize fewer process input resources than the available ones, and this is one more difference between this framework and its predecessor by Vergidis (2008). In this way, the framework has managed to find the most optimal process design literally.

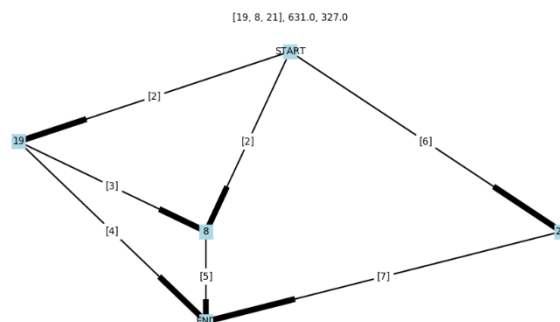


Figure 7.12 The optimal 3-task process design for Scenario A

7.3.2 Results for Scenario B

Figures 7.13, 7.14 and 7.15 show the results of eBPO_F for the sales forecasting process for each of the optimization algorithms. All EMOAs identify feasible solutions in all islands, the best of which shape the Pareto front. This reveals the workings of the new framework and the performance of the employed algorithms. Again, the optimization criteria are the minimization of the Service Delivery Price and the Service Fulfilment Target.

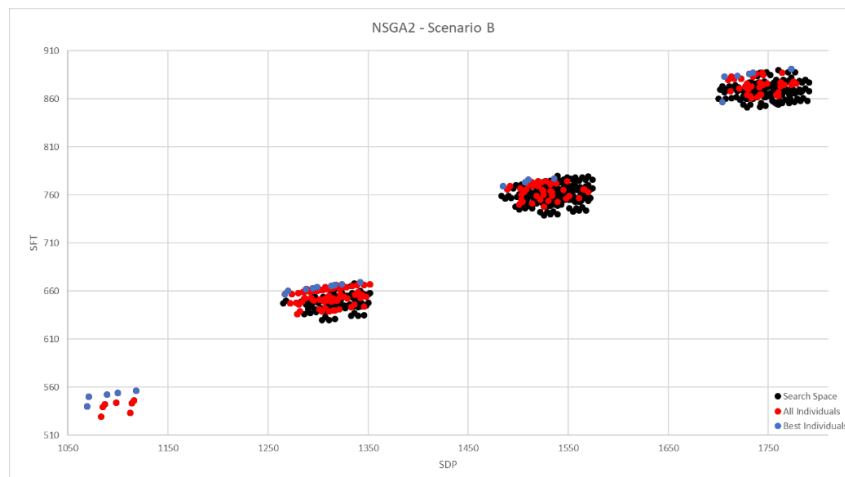


Figure 7.13 Scenario B under NSGA2

For this scenario, all invalid tasks but one after optimization occur from the “Check Task Inputs” sub process (chapter 4) and for this reason, no significant changes are observed to the search space since those tasks couldn’t be part of a feasible process design. In addition, the range of the attribute values is too narrow leading to a small search space for each island. Therefore, the feasible solutions found by all EMOAs are spread out all over the search space of each island but as it shown in the corresponding diagrams, they are definitely directed to the Pareto-optimal front.

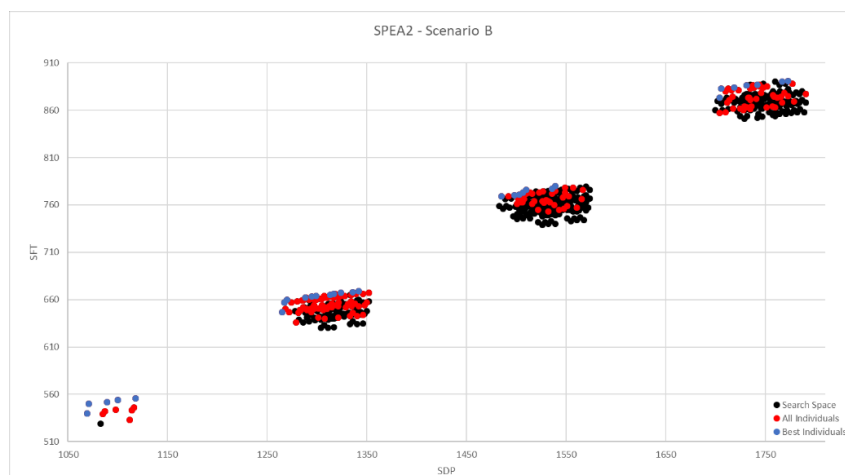


Figure 7.14 Scenario B under SPEA2

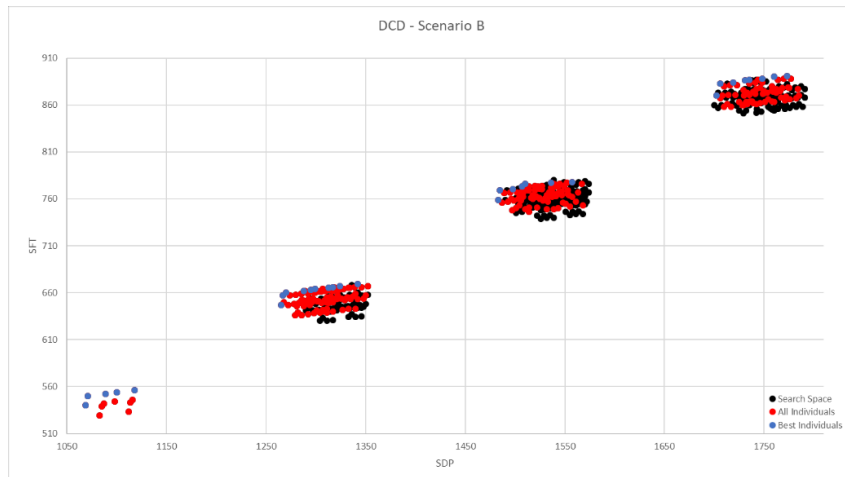


Figure 7.15 Scenario B under DCD

Figure 7.16 demonstrates two optimized business process designs for Scenario B. Both solutions have been found by all EMOAs. The left design belongs to the leftmost island of solutions in the diagrams above and comprises 5 tasks. The right design consists of 6 tasks and belongs to the second island in the diagrams. The 5-task design slightly differs from the draft of figure 7.2 since step 2, Retrieve business financial information, is covered by more than one tasks, specifically the tasks 10 and 2. In addition, the steps 3 and 4, Create sales forecast and Generate graph(s) respectively, can be executed simultaneously by the task 19 and then results are faxed back to the user. In this case, the new framework offers an optimized process design with an as reduced as possible cost.

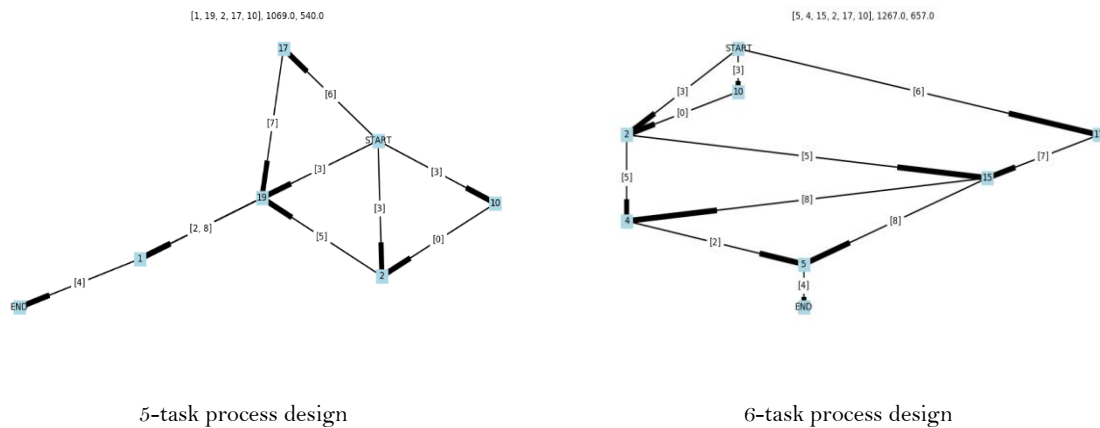


Figure 7.16 Optimized business process designs for Scenario B

On the other hand, in the right design of figure 7.16, the steps 3 and 4 are separated and their execution involves the tasks 15 and 4, respectively. This process design may provide more detailed graphs to the requested forecast report since the task 4 is exclusively a graph provider. Consequently, this alternative design increases the service level of the process in terms of providing finest visual information; it also increases the cost, though.

7.3.3 Results for Scenario C

Figures 7.17, 7.18 and 7.19 display the results of eBPO_F for the fraud investigation process for each of the EMOAs employed. All EMOAs identify feasible solutions in the first three islands and apparently, these solutions converge to and are spread out all over the Pareto-optimal front of those islands. The islands in the diagrams correspond to process designs with 3 to 7 tasks.

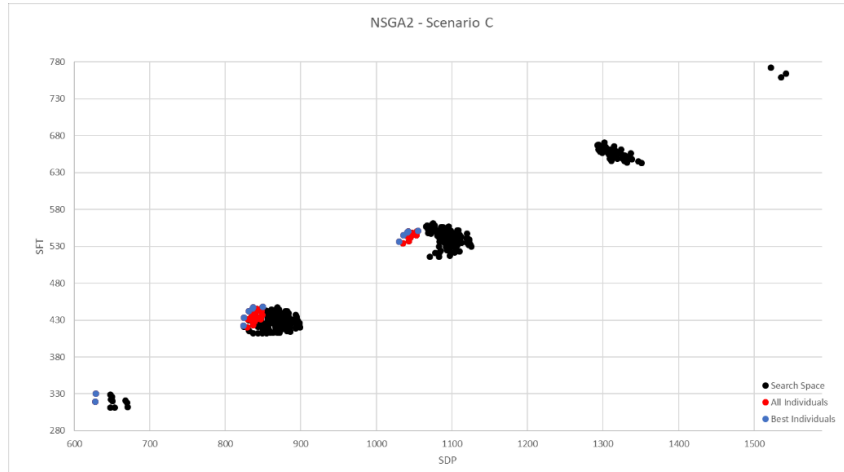


Figure 7.17 Scenario C under NSGA2

All EMOAs fail to find feasible solutions of the two rightmost islands because of the remaining valid tasks in the library after pre-processing. All invalid tasks after pre-processing for Scenario C, result from the “Check Categories” sub-process (chapter 4). This sub-process solely depends on the optimization type of the examined problem. In addition, as presented in a previous section, the decrease of the library size reaches 68%. This means that after pre-processing only few tasks remain in each of the categories. Hence, there are less available tasks to form OR patterns for particular resources and make the composed design more elaborated.

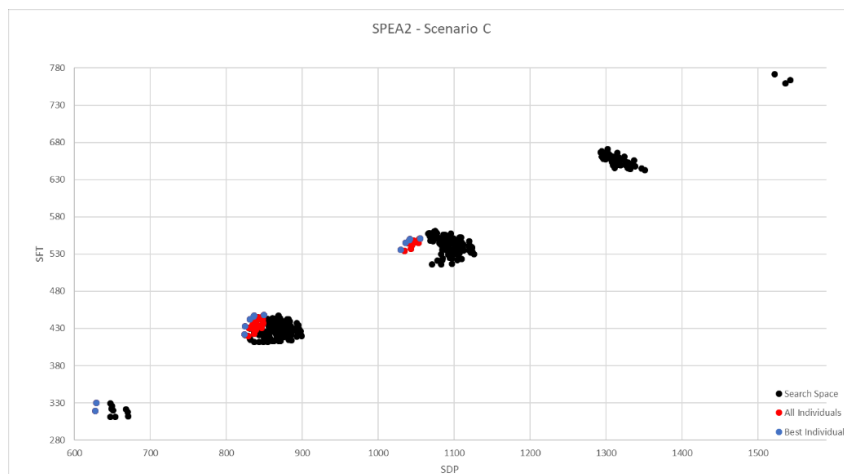


Figure 7.18 Scenario C under SPEA2

Furthermore, this must be also the case for two other optimization types, MIN-MIN and MAX-MIN. The range of the attribute values are too narrow leading to a small search space for each

island that is somewhat balanced and rounded. Consequently, without pre-processing, the central points of each island must follow a linear equation. However, as it is shown in the diagrams for Scenario C, in case of pre-processing and MIN-MIN optimization, the remaining valid tasks suffice only for designs with 3 to 4 tasks, hence the “gap” in the left bottom corner of the third island. In case of pre-processing and MAX-MIN optimization, eBPO_F cannot find solutions with more than 3 tasks. On the contrary, in case of pre-processing and MAX-MAX optimization, the new framework can find even more elaborated process designs for Scenario C. All things considered, in cases where the pre-processing of the library of tasks is very effective and the majority of invalid tasks result from the “Check Categories” subprocess, the optimization process is bounded to the literal optimized process designs regarding the problem type, and the capability of composing more elaborated designs, is considerably suppressed.

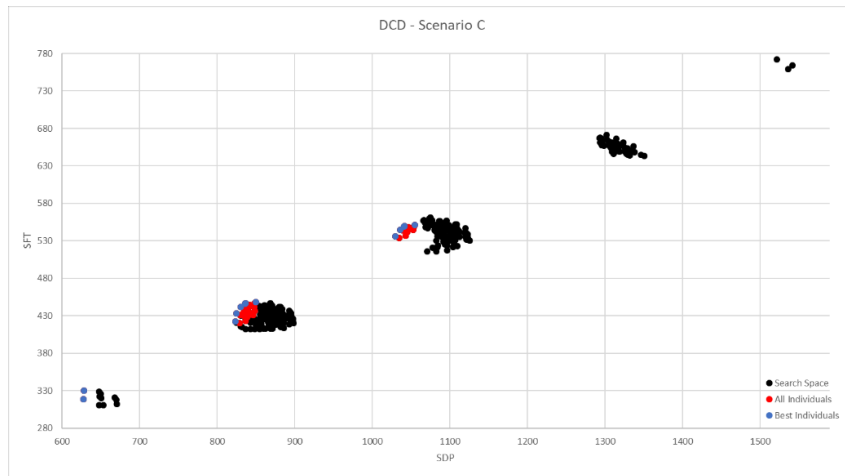


Figure 7.19 Scenario C under DCD

Figure 7.20 depicts two optimized business process designs for Scenario C. Both solutions have been found by all EMOAs. The left design of figure 7.20 comprises only 3 tasks and is literally one of the two most optimal solutions found for the fraud investigation process. Compared to the draft of figure 7.3, the task 6 can verify both the customer’s id and his credit card in order for the risk assessment report to be compiled.

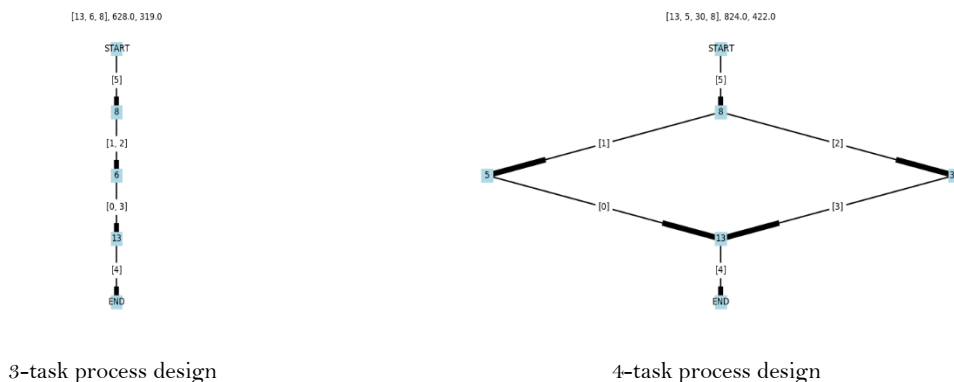


Figure 7.20 Optimized business process designs for Scenario C

On the other hand, the right design of figure 7.20, has exactly the same steps with the draft of figure 7.3 for Scenario C. The task 5 is responsible for the security login, the task 30 verifies the customer's id, the task 5 verifies the customer's credit card and finally, the task 13 compiles the risk assessment report.

7.4 Summary

This chapter presented the validation testing of the proposed optimization framework. Three real-life scenarios of business process designs have been examined. These scenarios were developed by Vergidis (2008) for the validation testing of BPO_F and were selected considering three different classifications in terms of business process automation. The proposed pre-processing technique has been tested as a standalone procedure and has demonstrated satisfactory results, managing to reduce drastically the problem dataset of all scenarios examined. In addition, the results of eBPO_F for the three real-life scenarios indicate that the framework can automate the process composition and identify alternative business process designs with optimized attribute values. The scenarios that already involved automated processes, focused on the optimization capability while the manual process scenario displayed that the new framework can enhance the process operation itself. The next chapter provides an overview of this research and a critical discussion on the limitations, the contribution and the potential for further research.

CHAPTER 8

Discussion & Conclusions

This chapter concludes this thesis and provides an overview of the main outcomes and the research contributions. In addition, the limitations of this research are discussed along with the issues that could have a potential for future work.

8.1 Thesis Overview

The aim of this thesis as stated in chapter 3, is the improvement and the extension of previous approaches for *Evolutionary Multi-objective Business Process Optimization*. The previous approaches that formed the basis for this thesis are owed to Hofacker and Vetschera (2001) and Vergidis (2008).

Hofacker and Vetschera (2001) were the first who attempted to optimize the design of (mainly administrative) business processes and applied evolutionary algorithms to the business optimization problem. However, the testing of their approach showed weak performance and they attributed it to the operations of the genetic algorithms since they could not maintain the feasibility of a design despite the fact that they incorporated the feasibility maintenance via a penalty term added to the objective function.

Vergidis (2008) was mainly involved with the optimization of business processes of the service industry and implemented in Java a *business process optimization framework* (BPO_F) to capture, visualize and express a business process design in a quantitative way that allows Evolutionary Multi-Objective Optimization Algorithms (EMOAs) to generate a series of alternative optimized designs. Based on the previously described approach, he managed to tackle the issue regarding the feasibility maintenance by introducing the *process composition algorithm* (PCA). PCA was responsible for the evaluation of each solution in the population. Given the quantitative representation of a solution, PCA tried to compose a business process design and checked if the result corresponded to a feasible design. While checking the feasibility of the produced design, PCA calculated the *Degree of Infeasibility* (DoI) of the solution examined. DoI measured the extent to which a process design is infeasible and the solutions that would form the parent population in each generation, would be those having the smaller DoI. In this way, DoI helped in selecting the

'less' infeasible solutions and preserving them in the population with the hope that have a better change of evolving towards feasible solutions during the optimization process and thus, the feasibility could be efficiently maintained.

The outcome of this research is a revised and improved version of BPO_F, the *extended business process optimization framework* (eBPO_F). The new framework incorporates a pre-processing technique for enhancing the efficiency of the employed Evolutionary Multi-objective Optimization Algorithms (EMOAs), a new process composition algorithm, PCA-II, suitable for real-world problems and many other features such as ease of use, more efficient I/O, better interactivity and easy maintenance.

Consequently, the first objective of this research was to study and understand the approaches described above. This objective has been carried out in chapter 2 where the literature review is presented. That chapter provided an overview of the most common definitions for business processes, presented the main business process modelling techniques and discussed the aforementioned optimization approaches in detail. The second objective was the review of BPO_F since it displayed satisfactory results with problems derived from the service industry and formed the basis for the new framework developed in this research. This objective would facilitate the author to identify the room for improvement.

After reviewing BPO_F, the third objective of this thesis was the refined and improved implementation of the existing framework in Python towards usability, I/O, interactivity and code maintenance. In addition, eBPO_F, the new software tool, enhanced the reviewing of the produced optimized business process diagrams through the production of graphics that the reviewer will be able to interact with. The functional overview and the I/O of eBPO_F along with the Python libraries used for implementing it, are presented in chapter 6.

The fourth objective of this thesis was also extracted from the review of eBPO_F. This objective involved the introduction of a pre-processing methodology for business process optimization (BPO) problems, aiming for increasing the ability of the EMOAs employed to find the optimized design alternatives. The business process multi-objective optimization problems belong to the NP-hard problems, which indicates that both the efficiency of optimization algorithms and the quality of produced results rely upon the size of the examined problem. In general, business processes have many available alternatives for the participating tasks and these, in turn, can involve many different resources as either requirements or products of their utilization. The improvement of the proposed pre-processing technique derives from clearing the problem dataset out of tasks that will never be part of the best solution or tasks whose features don't comply with

the problem constraints. The main steps of the pre-processing phase and the necessity behind such a technique in BPO are discussed in chapter 4.

The next objective of this thesis was the introduction of PCA-II, a new process composition algorithm that gives the ability to the new framework to solve more real-life BPO problems. Within the validation testing of the proposed pre-processing methodology, inconsistencies occurred between the results of the framework before and after pre-processing. As discussed in detail in chapter 5 where the PCA-II is presented, the pareto front of the solutions after pre-processing was always worse in all runs conducted. This contradiction is owed to the differences between the author of this thesis and Vergidis (2008), on what makes a process design perceived as feasible. The pre-processing algorithm has been designed by the author of this thesis with the intention to be generic enough to be applicable to any problem dataset. Therefore, the considered constraints for the feasibility of a business process design should be well-rounded. On the other hand, the problems that Vergidis (2008) was involved with, came from the service industry and PCA in his research was designed considering the needs of that domain. The approach of PCA itself raised some inherent limitations that didn't allow the framework to produce feasible solutions for more complex problems. PCA-II follows a different composition approach and uses a different business process representation which enable the new framework to meet the requirements of real-life business process, and EMOAs to be applied more effectively in BPO problems.

The final objective of this thesis was the validation testing of the revised optimization framework and the evaluation of the results. A set of real-life scenarios introduced by Vergidis (2008) was tested, but this time there were more constraints for a business process design to be feasible. In this way, the validity of the results could be achieved through the direct comparison of them against those produced by BPO_F . The proposed pre-processing technique was tested as a standalone procedure and demonstrated satisfactory results, managing to reduce drastically the problem dataset of all scenarios examined. Finally, the optimization results of $eBPO_F$ for the three real-life scenarios were very promising and indicated that the framework work as expected. It can automate the process composition and identify alternative business process designs with optimized attribute values.

8.2 Research Contribution

The contribution of this research to the domain of *Business Process Optimization* can be divided into three directions. The first one has to do with the introduction of a new software tool for BPO problems. This tool constitutes the improvement of the existing one towards usability, I/O, interactivity and maintenance. Its code is separated in modules and the best practises have been

considered throughout the development. At the end of the execution, the user will be able to see a graphic with the solutions found by the tool and interact with them to see the process diagram and compare different designs.

The second contribution to the domain of BPO is the proposal of a pre-processing algorithm, designed with the intention to be integrated in any BPO framework. This algorithm aims to improve the efficiency of the evolutionary algorithms so as for them to produce better solutions. The improvement derives from clearing the problem dataset out of tasks that will never be part of the best solution or tasks whose features don't comply with the problem constraints. Another gain from such a pre-processing algorithm is that the tool can work efficiently without the need of a problem-specific dataset. This way, it can be used as a mining tool for business processes that could be formed by already existing tasks; it should only be fed with the dataset containing all existing tasks in the business. In this case, the pre-processing stage can be used as a fast way to see if the given library of tasks is suitable for producing at least one feasible business process design according to the problem specification without executing the main operation of the framework.

The last contribution of this research is the new process composition algorithm for BPO problems. This algorithm gives the ability to the new framework to handle more complex problems since it follows a different approach for process composition than its predecessor by Vergidis (2008). It aims to capture the needs of a wide range of business process domains so that the framework will be capable of processes beyond the service industry. In addition, the proposed business process representation accompanying PCA-II, enables EMOAs to be applied more effectively in BPO problems.

8.3 Research Limitations

Although the feeling after completing this thesis is very hopeful since a new methodology on multi-objective optimization of business has been implemented as a software tool in Python, it would be improper not to unveil some of its limitations that have been identified during the process.

The first limitation comes from the pre-processing technique itself. In cases where the pre-processing of the library of tasks is very effective and the majority of invalid tasks result from clearing out the dominated tasks according to the problem type, the optimization process is bounded to the literal optimized process designs regarding the problem type, and the capability of composing more elaborated designs, is considerably suppressed. This may be undesired in business processes whose one of the optimization objectives is the Service Level. In such cases, a

business process may need to employ more than one provider for a particular resource making the process more reliable.

The second limitation originates from the crossover operator used by the proposed optimization framework since it doesn't ensure that the offspring cannot end up containing duplicate tasks. Duplicate tasks are not allowed in a process design composed by PCA-II, and this feature of the used crossover operator may also suppress the capability of composing more elaborated designs.

Another limitation is pinpointed to the fact that there is a narrow range of business process optimization problems in literature. Furthermore, the problem dataset of those problems is too small and doesn't capture that complex processes so as to be used for the performance evaluating of a business optimization framework.

Finally, the last limitation of the project has to do with the optimization of business processes whose one of the optimization objectives is the Process Duration. This objective usually depends on the joint presence of activities in a model, hence you cannot simply evaluate them individually because of the synergy effects. There is no assumption in the proposed pre-processing technique considering the needs of such an objective and PCA-II cannot accurately evaluate a composed process design regarding time duration, as is.

8.4 Future work

In light of the potential and the advantages of this project and the limitations identified during the process, there is a number of issues that could be potentially explored. Future research on the area of business process optimization could focus on developing a library of business process optimization problems, alleviating the crossover operator misbehavior and enabling the proposed optimization framework to deal with the time duration of a business process as an optimization objective.

The developing of a library of business process optimization problems would be a common point of reference and comparison for the validation and the performance evaluation testing of newly proposed optimization approaches. This library should include processes from a wide range of business domains with different levels of complexity.

The alleviation of having duplicate tasks in a solution because of the crossover operation would be a possible extension of the proposed optimization framework in this research. An additional contribution to DoI would be considered for the presence of duplicate tasks in the solution or another crossover operator could be either selected or developed that follows a different approach not allowing duplicate tasks in the offspring.

Finally, serious work should be made in the future for enabling the proposed optimization framework to deal with the time duration of a business process as an optimization objective. PCA-II should incorporate an additional assessment approach when dealing with such objectives where synergy effects are present. It may involve an additional factor in the objective function that should be considered for capturing and representing quantitatively those synergy effects in the process design.

8.5 Conclusion

This thesis improved and extended previous approaches for Evolutionary Multi-objective Optimization of business processes by providing a revised and refined version of an existing business process optimization framework that incorporates a pre-processing technique for enhancing the efficiency of the employed Evolutionary Multi-objective Optimization Algorithms (EMOAs), a new process composition algorithm that make the new framework capable of fulfilling more real-life constraints and handling more complex problems and many other features such as ease of use, more efficient I/O, better interactivity and easy maintenance. The proposed pre-processing technique was tested as a standalone procedure and demonstrated satisfactory results, managing to reduce drastically the problem dataset of all scenarios examined. The results of the whole optimization framework for the real-life scenarios examined, were very promising and indicated that the framework can automate the process composition and identify alternative business process designs with optimized attribute values.

References

- Abate, A.F., Esposito, A., Grieco, N. and Nota, G. (2002), 'Workflow Performance Evaluation Through WPQL', in *Proceeding of the 14th International Conference on Software Engineering and Knowledge Engineering Vol. 27*, ACM Press, New York, pp. 489-495.
- Agerfalk, P., Goldkuhl, G. and Cronholm, S. (1999), 'Information Systems Actability Engineering - Integrating Analysis of Business Process and Usability Requirements', in *Proceedings of the 4th International Workshop on the Language Action Perspective on Communication Modelling* Copenhagen, Denmark, pp. 1245-1252.
- Aguilar-Saven, R.S. (2004), 'Business Process Modelling: Review and Framework', *International Journal of Production Economics*, Vol. 90, pp. 129-149.
- Ahmadikatouli, A., and Aboutalebi, M. (2011), 'New Evolutionary Approach to Business Process Model Optimization', *Lecture Notes in Engineering and Computer Science*, Vol.2.
- Biazzo, S. (2002), 'Process Mapping Techniques and Organizational Analysis', *Business Process Management Journal*, Vol. 8, No. 1, pp. 42-52.
- Castellanos, M., Casati, F., Umeshwar, D. and Ming-Chien, S. (2004), 'A Comprehensive and Automated Approach to Intelligent Business Processes Execution Analysis', *Distributed and Parallel Databases*, Vol. 16, pp. 1-35.
- Coello Coello, C.A. (2005), 'Recent Trends in Evolutionary Multi-objective Optimization.', in Abraham, A., Jain, L. and Goldberg, R. (editors), *Evolutionary Multi-objective Optimization: Theoretical Advances and Applications*, Springer-Verlag, London, pp. 7-53.
- Davenport, T.H. (1993), *Process Innovation: Reengineering Work Through Information Technology*, Harvard Business School Press, Boston.
- Davenport, T.H. and Short, J.E. (1990), 'The New Industrial Engineering: Information Technology and Business Process Redesign', *Sloan Management Review*, pp. 11-27.
- Deb, K. (2001), *Multi-objective Optimisation Using Evolutionary Algorithms*, John Wiley & Sons, New York.
- Deb, K, Agrawal, S., and Meyarivan, T. (2000), 'A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-objective Optimization: NSGA-II', *Parallel Problem Solving from Nature - PPSN VI*. Lecture Notes in Computer Science, vol 1917. Springer, Berlin, Heidelberg
- Fan, Y.S. (2001), *Fundamental of Workflow Management Technology*, Springer-Verlag, New York.
- Georgoulakos, K., Vergidis, K., Tsakalidis, G., and Samaras, N. (2017), 'Evolutionary Multi-Objective Optimization of business process designs with pre-processing', *2017 IEEE Congress on Evolutionary*

Computation, pp. 897-904.

Goldberg, D.E., and Deb, K. (1991), 'A Comparative Analysis of Selection Schemes Used in Genetic Algorithms', *Foundations of Genetic Algorithms*, Vol. 1, pp. 69-93.

Grigori, D., Casati, F., Castellanos, M., Dayal, U., Sayal, M. and Shan, M.-C. (2004), 'Business Process Intelligence', *Computers in Industry*, Vol. 53, pp. 321-343.

Guha, S., Kettinger, W.J. and Teng, J.T.C. (1993), 'Business Process Reengineering: Building a Comprehensive Methodology', *Information Systems Management*, Vol. 10, pp. 13-22.

Gunasekaran, A. and Kobu, B. (2002), 'Modelling and Analysis of Business Process Reengineering', *International Journal of Production Research*, Vol. 40, No. 11, pp. 2521-2546.

Hammer, M. and Champy, J. (1993), *Reengineering the Corporation: A Manifesto for Business Revolution*, N. Brealey, London.

Havey, M. (2005), *Essential Business Process Modelling*, O'Reilly, U.S.A.

Hofacker, I. and Vetschera, R. (2001), 'Algorithmical Approaches to Business Process Design', *Computers & Operations Research*, Vol. 28, pp. 1253-1275.

Irani, Z., Hlupic, V. and Giaglis, G.M. (2002), 'Business Process Reengineering: An Analysis Perspective', *International Journal of Flexible Manufacturing Systems*, Vol. 14, pp. 5-10.

Johanson, H.J., McHugh, P., Pendlebury, A.J. and Wheeler, W.A.I. (1993), *Business Process Reengineering - Breakpoint Strategies for Market Dominance*, Wiley, Chichester.

Kim, C.H., Weston, R.H., Hodgson, A. and Lee, K.H. (2003), 'The Complementary Use of IDEF and UML Modelling Approaches', *Computers in Industry*, Vol. 50, pp. 35-56.

Koubarakis, M. and Plexousakis, D. (2002), 'A Formal Framework for Business Process Modelling and Design', *Information Systems*, Vol. 27, pp. 299-319.

Lakin, R., Capon, N. and Botten, N. (1996), 'BPR Enabling Software for the Financial Services Industry', *Management Services*, Vol. 40, pp. 18-20.

Lindsay, A., Downs, D. and Lunn, K. (2003), 'Business Processes - Attempts to Find a Definition', *Information and Software Technology*, Vol. 45, pp. 1015-1019.

Luttighuis, P. O., Lankhorst, M., van de Wetering, R., Bal, R., and van den Berg, H. (2001), 'Visualizing Business Processes', *Computer Languages*, Vol. 27, pp. 39-59.

Melao, N. and Pidd, M. (2000), 'A Conceptual Framework for Understanding Business Process Modelling', *Information Systems*, Vol. 10, pp. 105-129.

Miller, B.L., and Goldberg, D.E. (1995), 'Genetic Algorithms, Tournament Selection, and the Effects of Noise', *Complex Systems*, Vol. 9, No. 2, pp. 193-212.

Pall, G.A. (1987), *Quality Process Management*, Prentice-Hall, Englewood Cliffs, NJ, USA.

- Sadiq, W. and Orłowska, M. (2000), 'Analyzing Process Models Using Graph Reduction Techniques', *Information Systems*, Vol. 25, No. 2, pp. 117-134.
- Shen, H., Wall, B., Zaremba, M., Chen, Y. and Browne, J. (2004), 'Integration of Business Modelling Methods for Enterprise Information System Analysis and User Requirements Gathering', *Computers in Industry*, Vol. 54, pp. 307-323.
- Smith, H. (2003), 'Business Process Management - the Third Wave: Business Process Modelling Language and Its Pi-Calculus Format', *Information and Software Technology*, Vol. 45, pp. 1065-1069.
- Soliman, F. (1998), 'Optimum Level of Process Mapping and Least Cost Business Process Re-Engineering', *International Journal of Operations and Production Management*, Vol. 18, No. 9/10, pp. 810-816.
- Stock, J.R. and Lambert, D.M. (2001), *Strategic Logistics Management*, McGraw-Hill, Irwin.
- Stohr, E.A. and Zhao, J.L. (2001), 'Workflow Automation: Overview and Research Issues', *Information Systems Frontiers*, Vol. 3, No. 3, pp. 281-296.
- Tinnila, M. (1995), 'Strategic Perspective to Business Process Redesign', *Business Process Re-Engineering and Management Journal*, Vol. 1, No. 1, pp. 44-59.
- Tiwari, A. (2001), 'Evolutionary computing techniques for handling variables interaction in engineering design optimisation'. PhD Thesis, SIMS, Cranfield University, Cranfield, U.K.
- Tiwari, A., Vergidis, K. and Majeed, B. (2006), 'Evolutionary Multi-Objective Optimisation of Business Processes', in *Proceedings of IEEE Congress on Evolutionary Computation 2006* Vancouver, Canada, pp. 3091-3097.
- Tiwari, A., Vergidis, K. and Roy, R. (2007), 'Evolutionary Optimization of Business Process Designs', *Evolutionary Scheduling*, Studies in Computational Intelligence, vol 49. Springer, Berlin, Heidelberg.
- Tiwari, A., Turner, C., Ball, P., and Vergidis, K. (2010), 'Multi-Objective Optimisation of Web Business Processes', *Simulated Evolution and Learning*. Lecture Notes in Computer Science, vol 6457. Springer, Berlin, Heidelberg.
- van der Aalst, W.M.P., ter Hofstede, A.H.M. and Weske, M. (2003), 'Business Process Management: A Survey', *Lecture Notes in Computer Science*, Vol. 2678, pp. 1-12.
- Vergidis, K., Turner, C., Alechnovic, A., and Tiwari, A. (2015), 'An automated optimization framework for the development of re-configurable business processes: a web services approach', *International Journal of Computer Integrated Manufacturing*, Vol.28, pp. 41-58.
- Vergidis, K. (2008), 'Business Process Optimisation using an Evolutionary Multi-Objective Framework.' PhD Thesis, Cranfield University, Cranfield, U.K.
- Vergidis, K., and Tiwari, A. (2008), 'Business process design and attribute optimization within an evolutionary framework', *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, pp. 668-675.
- Vergidis, K., Tiwari, A. and Majeed, B. (2007), 'Business Process Analysis and Optimization: Beyond

- Reengineering', *IEEE Transactions on Systems, Man, and Cybernetics—Part C: Applications and Reviews*, Vol. 38, pp. 69-82.
- Vergidis, K., Tiwari, A., Majeed, B., and Roy, R. (2007), 'Optimisation of business process designs: An algorithmic approach with multiple objectives', *International Journal of Production Economics*, Vol. 109, pp. 105-121.
- Vergidis, K., Tiwari, A. and Majeed, B. (2007), 'Composite business processes: An evolutionary multi-objective optimization approach', *2007 IEEE Congress on Evolutionary Computation*, pp. 2672-2678.
- Vergidis, K., Tiwari, A. and Majeed, B. (2006), 'Business Process Improvement Using Multi-Objective Optimisation', *BT Technology Journal*, Vol. 24, No. 2, pp. 229-235.
- Volkner, P. and Werners, B. (2000), 'A Decision Support System for Business Process Planning', *European Journal of Operational Research*, Vol. 125, pp. 633-647.
- Wang, K., Salhi, A. and Fraga, E.S. (2004), 'Process Design Optimisation Using Embedded Hybrid Visualisation and Data Analysis Techniques Within a Genetic Algorithm Optimisation Framework', *Chemical Engineering and Processing*, Vol. 43, pp. 663-675.
- Wang, M. and Wang, H. (2005), 'Intelligent Agent Supported Business Process Management', in *Proceedings of the 38th Hawaii International Conference on System Sciences*, Hawaii, pp. 567-577.
- Zakarian, A. (2001), 'Analysis of Process Models: A Fuzzy Logic Approach', *International Journal of Advanced Manufacturing Technology*, Vol. 17, pp. 444-452.
- Zakarian, A. and Kusiak, A. (2001), 'Process Analysis and Reengineering', *Computers & Industrial Engineering*, Vol. 41, pp. 135-150.
- Zitzler, E., Laumanns, M. and Thiele, L. (2001), 'SPEA2: Improving the Strength Pareto Evolutionary Algorithm', *TIK Report Nr. 103*, Computer Engineering and Networks Lab (TIK), Swiss Federal Institute of Technology (ETH) Zurich.
- Zitzler, E., and Thiele, L. (1999), 'Multi-objective evolutionary algorithms: A comparative case study and the strength pareto approach', *IEEE Transactions on Evolutionary Computation*, Vol.3, pp. 257-271.