Autonomic Business Processes

Paul Nicholas Taylor

Doctor of Engineering
University of York
Computer Science

April 2015

Abstract

Business processes in large organisations are typically poorly understood and complex in structure. Adapting such a business process to changing internal and external conditions requires costly and time consuming investigative work and change management. In contrast autonomic systems are able to adapt to changing environments and continue to function without external intervention. Enabling business processes to adapt to changing conditions in the same way would be extremely valuable. This work investigates the potential to self-heal individual business process executions in generic business processes. Classical and Immune-inspired classification algorithms are tested for their predictive utility with Decision Trees augmented with MetaCost and Immunos 99 exhibiting the best performance respectively. An approach to deriving recovery strategies from historical process data in the absence of a process model is presented and tested for suitability. Also presented is an approach to selecting the best of the determined recovery strategies for application to a business process execution, which is then tested to determine the impact of its parameters on the quality of selected recoveries.

Contents

List of Figures

List of Tables

Preface

The research work for this thesis was undertaken while the author was employed at BT. As such the data sets used, and the models and results produced contain confidential information which cannot be included in a public document. To address the corporate confidentiality concern process specifics have been masked or only reported in general terms throughout.

Acknowledgements

Special thanks are due to my academic supervisors Dr. Fiona Polack and Prof. Jon Timmis. Their time, advice and guidance have been invaluable. It's been a pleasure to work with you both.

Thanks are also due to my industrial supervisor Dr. Detlef Nauck, both for his academic and industrial guidance, and for his seemingly inexhaustible supply of patience.

Declaration

Except where stated, all of the work contained within this thesis represents the original contribution of the author.

Work appearing within or closely related to work in this thesis has appeared as:

- P. Taylor *et al.*, ‘Case Study in Process Mining in a Multinational Enterprise’, in *Data-Driven Process Discovery and Analysis*, ser. Lecture Notes in Business Information Processing, K. Aberer *et al.*, Eds., vol. 116, Springer Berlin Heidelberg, 2012, pp. 134–153 [1].
- M. Spott *et al.*, ‘Modern Analytics in Field and Service Operations’, in *Transforming Field and Service Operations*, G. Owusu *et al.*, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, ch. Part II, pp. 85–99 [2].
- P. Taylor *et al.*, ‘Accelerating Immunos 99’, in *Advances in Artificial Life, ECAL*, vol. 12, MIT Press, Sep. 2013, pp. 893–898 [3].

Additionally the following reports were prepared for BT based on the work within this EngD to inform the BT’s Future Organisation strategic research programme.

- P. Taylor, ‘Autonomic Process Management for BT’, BT Innovate and Design, Internal Report, Mar. 2012 [4].
- P. Taylor, ‘Process Prediction for Openreach’, BT Research and Innovation, Internal Report, 2014 [5].
- P. Taylor, ‘Autonomic process business impact model’, BT Technology, Service and Operations, Internal Report, 2014 [6].

This thesis document will also be contributed to the BT’s Future Organisation research programme for 2015/2016 upon publication.

Chapter 1

Introduction

In the commercial world business improvement is extremely important. By producing products faster, cheaper, and in larger quantities a business can become more profitable. Accordingly there is huge interest in business improvement techniques and production methodologies that can enable this to occur (e.g. [7]–[9]). In general, techniques and methodologies for business improvement deal with first understanding and then controlling elements of the production process and in turn the process itself.

Technical contributions to business improvement fall into two main groups, tools to orchestrate or control the business process (e.g. business process management systems such as [10] or [11]) and tools to measure business performance (e.g. Dashboards like Oracle Business Intelligence [12] or Statistical Process Control [13]). A process which was able to adapt to changes in the business environment would reduce the time and effort taken to monitor and improve the processes manually, which would lead to decreased costs through the reduction of process improvement programmes.

One increasingly popular tool to inform process improvement is that of process mining [14]. Process mining seeks to reconstruct the business process from the execution logs that are stored by workflow systems. Referring to a model derived from real data ensures the improvements are grounded on that basis, rather than on the mental model of the process improvement consultant. Applying process mining to large scale business processes is not without its own problems. In many cases there is a conceptual mismatch between process designers, operational staff and IT systems engineers involved in orchestrating a business process. This disconnect is particularly problematic for process mining [1].

One view of a business process is to see it as the software that runs on top of the hardware of the business to produce the output required. Therefore approaches to organising computer systems may provide some inspiration for improving business processes.

In academia there is a significant amount of research being conducted into the self-organisation of systems in order to bring into software the ability to self-configure, self-heal, self-optimise and self-protect [15], also known as self-* properties. These autonomic systems are able adapt to changes in their execution environment and continue to function without explicit reprogramming or reorganisation. The autonomic properties are ideal properties of business processes.

Autonomic approaches are not entirely without precedent in industry, products incorporating self-* principles have been developed to ease the monitoring and configuration workload for data centres and networks in products such as IBM's autonomic computing toolkit [16]. While the autonomic responses of these products make the infrastructure of the business more robust the autonomic principles are not applied to the actual business processes and tasks utilising the infrastructure. The existing approaches and products are able to ease the burden on system and network administrators caused by changes and failures. Easing the burden on infrastructure administration staff allows them to concentrate more time on the development of the estate and on the delivery of new work rather than them needing to spend the bulk of their time on problem management and fire-fighting.

1.1 Business Processes

Considering the operation of a business as a process, or a series of processes, is not a recent insight. Indeed Adam Smith's 1776 work "*An Inquiry into the Nature and Causes of the Wealth of Nations*" [17] alludes to a business process in a description of the manufacturing process for making pins. The organisation of the production and the organisation of the flow of pins between different people is analogous to a business process. Each worker in the factory was assigned a task to perform and the sequence of tasks required to produce a pin was determined. Defining and following a process allowed pins to be manufactured at a greater rate per person employed than had been possible with an identical number of individual pin makers.

In the literature there are a number of definitions of a business process. The standard definition used is usually either that of Hammer and Champy [18] or that of Davenport [19].

Hammer and Champy define a business process as *"a set of activities that, taken together, produce a result of value to a customer"* [18, p. 3]. The strength of this definition is that it does not constrain the types of activities or the types of actors that might be involved in the execution of the process. The major disadvantage of the definition is that it is limited in scope to only those processes which create a customer output, which would preclude the notion of internal processes. For example, promoting or hiring a new employee would not create an output for a customer but is an essential business process.

Davenport provides an alternate definition of a business process: *"A process is simply a structured, measured set of activities designed to produce a specified output for a particular customer or market"* [19, p. 5]. This more comprehensive definition of a business process includes a more explicit mention of an individual customer or market which is an even stronger omission of non-customer-facing processes. Davenport's definition also includes an element of structure, and known input and output, which is appropriate for clean-room implementations but in practice it has been found that inputs/outputs are not well defined [1]. The cause of many process execution issues is the execution of transitions between tasks before the inputs are available or the outputs are finished.

While the precise definitions used vary, common themes are the collection of tasks, the production of a product or the output, and the organisation of the tasks.

In order to ensure a consistent basis for discussion on business processes we use the following definition incorporating the salient points. This definition is based on Hammer and Champy's [18] definition with the emphasis shifted from customer output to measurable output.

There should be some result of applying a business process within its business context. This result may be the production physical object in a manufacturing process, or an intangible one (eg. a firewall port being opened). If it is possible to determine if the output has been produced as desired then the output is measurable.

Definition: 1. *A business process is the organisation of defined actions that are required to be performed by the business in order achieve a measurable output.*

It is important to note that Definition 1 says nothing about what the actions should be or at which level they should be recorded. Figure 1.1 shows an example of a simple, high-level business process for resolving a fault report. It is easy to see that the process tasks could be decomposed into a more detailed design utilising both automated and manual activities across many different departments (or indeed different businesses spread across the globe).

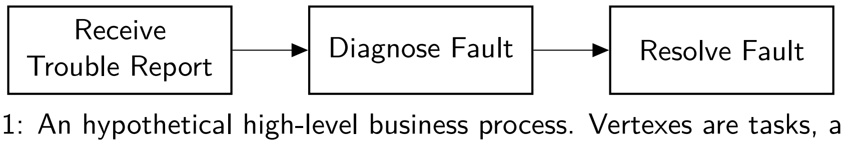

Figure 1.1: An hypothetical high-level business process. Vertexes are tasks, and edges are transitions between them.

There are many ways to decompose business processes from high level goals into specific activities, indeed this is the core of business process design. Many businesses will use a naive approach where interested parties meet face to face and design the process on paper by decomposing each task into sub-tasks until the level of individual activities is reached. A decomposition of the example business process in Figure 1.1 follows with each additional level of indentation representing an additional decomposition. The lowest level activities would form the business process to be implemented.

1. Receive Trouble Report
    (a) Customer contacts agent
    (b) Agent validates customer identity
    (c) Agent validates impacted service
    (d) Trouble ticket is created
2. Diagnose Fault
    (a) Automated diagnostics executed on impacted product
    (b) Perform further diagnostics if automated diagnostics are inconclusive
    (c) If issue cannot be diagnosed then escalate to product specialists
3. Resolve Fault
    (a) Implement the remedial actions
        - If an engineering visit is required
            i. Create trouble ticket with supplier
            ii. Reserve engineering appointment with customer
        - if an engineering visit is not required perform remedial actions
        - if remedial actions are unsuccessful return to 2.
    (b) Apply compensation to customer account for out of service time
    (c) Close trouble ticket.

Business processes are not constrained to a sequence. It is possible for multiple departments/businesses to each be performing work on a particular job simultaneously e.g. in a telecommunications context a supplier could be building and shipping customer premise equipment (CPE) while the provider is provisioning the service. Processes containing automated or robotic elements are not exempt from this, although at the industrial partner robotic tasks are usually executed in sequence. In this fashion a business process can be considered to be the definition (or expectation) of how the execution should occur.

Within the process mining community, each execution of an instance of a business process is known as a **Process Instance**. A process instance contains information on the tasks executed and the attributes used during the execution of the business process. Process attributes record information about the process such as its start and end time, and other domain specific information such as the applicable customer name. The task information in a process instance is characterised within BT in terms of task instances which record the executions of the individual tasks contained within the process instance.

Definition: 2. *A process instance is the execution of the tasks according to the structure and definitions in the applicable business process.*

A concept closely related to business process is workflow. The Workflow Management Coalition [20] defines a workflow as:

Definition: 3. *"The automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules"* [21, p. 8]

Definition 3 illustrates the link between workflow and business process. Business processes necessarily require humans or systems that effect the tasks. These participants are collectively referred to as actors. Business processes are concerned with the tasks (automatic or manual) and their execution; workflow is concerned with the transition of tasks between actors in the process. The relationship between workflow and business process is often not a 1:1 relationship as a single person may execute multiple tasks before passing a job onto the next stage in the process.

What is a task?

A canonical definition of a task or activity is not obvious. The difficulty in definition of a task comes from the fact that business processes can be and often are represented at different levels of detail [22]. Case studies and experience applying process mining in a large business has shown that real business processes can exist at a variety of different conceptual levels [1]. The following description is based upon work undertaken in the context of an international service provisioning process in the telecommunications domain.

At the highest level process designers typically view their processes as a series of abstract concepts which are linked in a prescribed way. Each of the elements at this conceptual level would likely be broken down into multiple tasks during the implementation of a process in the IT estate of the business. For example a concept might be "Provide Circuit from A-End to B-End". However this will require many tasks to be completed by many people. This level will be referred to as the conceptual level.

At the next level the operational managers that are responsible for all of, or part of, the process would see each separate piece of work that needed to be done as a task. Often the boundary between these tasks would be where the job must be passed from one team to another as new skills are required. This will be referred to as the operational level. The description of the production of a pin by Smith [17] is an example of an operational level description. Another example of an operational level description for providing a new telephony circuit to a customer might be:

Task 1: Validate customer address

Task 2: Validate network capacity is available

Task 3: Reserve required network elements

Task 4: Order Customer Premise Equipment (CPE)

Task 5: Arrange engineering appointment for service installation

Task 6: Receive notification from engineer that the service is installed

Task 7: Start customer billing

Task 8: Close order

The workflow level, the lowest level, would come from the people who manage the implementation of a process in the workflow systems of the business. Within the workflow system the conceptual or operational task being executed is largely irrelevant, all that is important is who should be working on the task now, and where it should go next. The transitions between actors that are finally defined at the workflow level are closely related to how the job should be passed from one team to another at the operational level, but the way this is organised may be unrelated to either the conceptual, or operational, design of the business process.

Two concepts closely related to workflow are those of queues and transitions. A queue is a container for work which is serviced by one or more actors. Each actor will take process items from the queue one at a time as per the requirements of the process execution. Once the actor has completed its processing it needs to move the job onto the next stage of the workflow. The movement of the process execution from one queue to the next is known as a transition.

A concrete example of multi-level design and implementation, based upon an extract of tasks from a real process in the telecommunications domain is given in Table 1.1.

At the conceptual level, the process in Table 1.1 consists of only two high-level tasks (Stage 2 and Stage 3) which are executed in sequence. At the operational level, the process contains 7 tasks and 6 transitions which are executed in sequence. Finally, at the workflow level the process utilises 4 queues and has 5 transitions.

Table 1.1: Illustration of the relationship between conceptual levels of business processes. Reading across the table increases the abstraction, reading down the table shows the sequence of actions. The Operational/Workflow levels show the non-1:1 mapping from business process to workflow. Task names and queue names have been redacted.

Conceptual Level	Operational Level	Workflow Level
Stage 2	Task A	Queue 101
	Task B	Queue 102
	Task C	Queue 101
Stage 3	Task D	Queue 103
	Task E	
	Task F	Queue 12
	Task G	Queue 102

1.2 Autonomic Systems

Autonomic (or self-* systems) are those which are able to respond to changes in their execution environment in such a way that they are able to continue to function, at either full capacity or with a known degradation, in spite of those changes in environment [23]. This thesis will consider how autonomic systems could be applied to the execution of business processes.

IBM, which incorporates some autonomic elements into some of its products, define four autonomic properties that can be exhibited by such a system: self-configuration, self-healing, self-optimisation and self-protection [23]. IBM's systems are typically given policies which guide the changes that the autonomic elements are able to make, and also provide guidance on which of a set of generated changes would be the most beneficial.

The self-* properties of autonomic systems have applications to business processes and their execution. Self-configuration refers to the ability of a system to adjust its executions to fit the current environment. Within business processes this could take the form of updating the selected tasks when new facilities are made available. For example, updating process executions to use a new line testing tool that has improved accuracy.

Self-healing refers to the ability of a system to recover from the failure of its components. Within business processes the main components are the tasks that are to be performed. An example of a self-healing response to a business process would be to re-route business processes around a task that is currently unavailable (perhaps due to underlying systems issues). The business process executions impacted would be healed to avoid the problem.

Self-optimisation is the process by which a system is able to tune its use of resources to best fit the system's overall goals. For example, the system may adjust the tasks it uses to prefer those with shorter execution times if the focus is on the rate that processes complete. Alternatively, the system might prefer higher cost but more accurate diagnostics when customer satisfaction takes priority over throughput. More usually there would need to be a balance between a number of completing priorities and the system would be able to optimise itself to a compromise solution.

Finally, self-protection is the ability of the system to discover and resist attacks on the system irrespective of the source. The nature of attacks on the system can be varied, in the business process domain social-engineering could be considered an attack on a customer interface procedure where an attack is trying to extract data from a business process. Within telecommunications, there is the concept of 'Slamming' where one company attempts to appropriate the lines of another, a service transfer process could be augmented with self-protection to try and prevent slamming from being successful.

1.3 Business Context

The industrial partner for this work is BT, a multinational telecommunications company with operations in over 170 countries around the world. As a large, very process driven company BT has a large amount of data available to trace the execution of a very large number of business processes. Processes running at high volume require automated solutions for problem detection and analysis because the number of people required to manually inspect each order would be prohibitively expensive.

These processes cannot run flawlessly at all times since not all eventualities and scenarios can be predicted ahead of time. Additionally, the regulatory and business framework can change which in turn has an impact of business processes. These issues and the impact of those on process analysis work are discussed in detail in [1].

A successful autonomic process system from a BT perspective is one which will allow it to autonomously correct business process issues and in turn reduce the impact of process failure on customers. Reducing the rate of process failure will lower the impact of that failure on customers, and should therefore result in an increase in customer satisfaction.

1.3.1 Classification Performance

A key performance indicator for the work in this thesis will be the performance of any predictive or classification algorithms that are used. In analysing the performance of these the primary measure of performance has not been the simple headline accuracy measure for each run. The true-positive rate for cases of predicted process failure is a better comparative measure from the perspective of businesses using these techniques. From a reputational or customer service viewpoint, performing extra work away from the customer on an order that would pass has no effect but failing to perform preventative work on an order that does fail gives a negative customer experience and requires additional corrective action. Misclassifying a process failure is therefore worse than misclassifying an process success.

Although a higher false positive rate for process failures is therefore acceptable, a decision would need to be made by the business as to what rate is appropriate for the situation. The acceptable false positive rate chosen would depend on how the organisation chooses to deal with predicted failures. If the business is using a highly trained specialist to analyse each of these, a significant manpower cost could be wasted investigating orders that would pass if the false positive rate was too high.

Acceptable values for the performance characteristics of the classifier are highly context dependent. Within the business context each of the four classification outcomes can be mapped onto a business scenario involving increasing or decreasing costs as follows:

- True Positive - This is a successful intervention so would usually reduce costs through the avoidance of failure and the associated recovery actions.

- False Positive - This is an unnecessary intervention where a process that would complete successfully has been identified as a failure. This will be a cost increase as the cost of intervention has been incurred.

- True Negative - This is a successful process execution that had no intervention. There is no change to costs for these process executions.

- False Negative - These are failures that have not been detected by the classifier. There is a cost impact on these processes involved in correcting the process and resulting issues. There is also an indirect cost to customer satisfaction since the customer has likely been inconvenienced by the failure.

Given these cost impacts it follows that the acceptable performance of a classifier needs to be informed by the cost impact of each of the classification outcomes and a balance struck. For example, a classifier with a True Positive Rate of 40% and a False Positive Rate of 1% is likely to yield significant cost savings to the business even though 40% is not a majority of the failures. This is acceptable because the cost-benefit of such a classifier is significant (assuming the cost savings for a true positive outweigh the cost incurred for a false positive). Of course, what counts a significant cost savings is also highly business and process dependent.

1.4 Research Hypothesis

All of the elements discussed in this chapter lead up to the formulation of the research hypothesis for this work. This will be the focal point for the conclusions in Chapter 7. The research hypothesis brings together the question that is to be answered, and runs as follows:

> Business process improvement can incorporate autonomic properties of self-healing to provide solutions to changing business contexts.

Each part of the thesis investigates a different facet of the above hypothesis. Each of these can be mapped to a more precise sub-hypothesis as follows:

Hypothesis 1:	Using historical business process data it is possible to predict the outcome of a business process using a classical classification technique (Chapter 3).
Hypothesis 2:	It is possible to produce a better performing classifier to predict the outcome of a business process using immune-inspired classification techniques rather than classical classification techniques (Chapter 4).
Hypothesis 3:	It is possible to produce accurate and useful recovery plans for business processes in the absence of a formal model of the business process (Chapter 5).

These hypotheses will be used to guide the conclusions at the end of the thesis.

1.5 Thesis Structure

The remaining chapters of this thesis are broken down to encompass specific elements of the research that was conducted.

Chapter 2 reports on a survey of literature appropriate to business processes and to self-* computing principles. The literature is surveyed with focus on applicability to the research hypothesis.

Chapter 3 investigates the appropriateness of classical machine learning classifiers for the prediction of business process outcomes. The chapter proposes a mechanism for creating data sets from business process data that can be processed for those classification tests. The performance of the classical classifiers for this prediction task is then examined.

Chapter 4 reports on testing that was performed to determine if classifiers using Artificial Immune Systems techniques could be used as a replacement for the classical techniques which would not require parameter tuning. A technique that could provide accurate predictions without requiring manual parameter tuning would vastly increase the speed with which a system implementing the techniques from this thesis could be deployed into new business processes. The chapter evaluates the results of experiments upon these classifiers and considers if they might be able to replace the classical classifiers for predicting business process outcomes.

Chapter 5 discusses how recovery strategies can be deduced from historical process data, and how a process monitoring system might be extended to include automatically determined recovery actions into an existing workflow. Experiments are performed using historical data to determine the performance of the proposed approach in both determining recovery strategies and in choosing the correct strategy for an intervention.

Chapter 6 evaluates the work undertaken in the thesis from scientific and industrial standpoints. Opportunities for further work that have been identified are also discussed.

Chapter 7 draws together and evidences conclusions drawn from throughout the thesis and identifies the novel contributions of this work.

1.6 Contribution to Knowledge and Business

This EngD project has investigated techniques for bringing self-healing features to generic business processes. These have been developed in conjunction with BT as the industrial partner for this work. Contributions to knowledge and to the business of the industrial partner are summarised below.

1.6.1 Contributions to Knowledge

The main contributions to knowledge made by this thesis are:

1. Propose a method for using historical process data and process mining techniques to produce classifiers suitable for the prediction of generic business processes. The method includes a description of what data to capture and how that data should be structured for classification (Section 3.1).

2. Propose the use of artificial immune systems techniques for the classification of business processes (Chapter 4). This is a new area of application for AIS

inspired classifiers. Four classifiers are tested with Immunos 99 showing the best performance for this scenario.

3. Propose and demonstrate a method for the discovery of recovery strategies for generic business processes without requiring an accurate model of the business process (Section 5.2). This method utilises data on historical executions of the process and the classifiers discussed in Chapters 3 and 4 to produce a recovery database of the recovery strategies and some measures of their historical success. The proposed method is classifier agnostic.

4. Propose a method for the selection of the most appropriate recovery strategy for the current business process state suitable for generic business processes (Section 5.3). This would allow an self-healing business process system to select an appropriate recovery path for a detected and impending business process failure. This method is also agnostic to the methods used to classify the business processes.

5. Propose a systems architecture that could be adopted by a business to implement the self-healing techniques contributed by this thesis in an existing business process system. The proposed system ensures clean separation and simple interfacing between the existing workflow systems of the business and the self-healing system (Section 5.3).

1.6.2 Contribution to Business

In addition to those contributions that this thesis makes to academic knowledge the thesis makes a number of contributions to the industrial partner. The exact impacts on the business of these contributions are commercially sensitive.

1. Investigated artificial immune inspired approaches for classification. In addition to the academic contribution of this work, this introduces a new set of techniques to BT that can be used an a variety of other areas, such as fault classification, churn prediction and alike.

2. Studied in detail a specific BT process to inform the business on the near-term suitability of the Immunos 99 for business process classification (Section 4.6). This enables the business to be more confident that the results presented in the thesis are applicable to real BT processes.

3. Proposed an end-to-end system for the self-healing of business processes that can be used to augment any existing workflow system. The proposed system would minimise the risk to BT's existing systems since it requires minimal new interfaces

to be effective and could be implemented in phases from detection first to complete unattended recovery at the end (Section 5.3).

4. Performed a cost-benefit analysis of the potential impact of a self-healing process system on failure rate and cost for business processes. This is a key way to measure the potential impact of a new system or approach (Section 6.2.1). The results of the cost benefit analysis show the potential impact of such a self-healing system and help to prove the case for continued research in this area.

Chapter 2

Literature Review

In this chapter literature relevant to the thesis is reviewed. Section 2.1 reviews the literature on concepts of business processes and business improvement. Section 2.2 discusses existing approaches for bringing the self-* properties to business processes and discusses the applicability of each approach.

2.1 Business Processes

2.1.1 Working with Business Processes - Design and Improvement

A simple way of designing a business process in a greenfield situation would be to gather together knowledgeable users from the interested parts of the business and use their expertise and discussion to generate an appropriate process design. This simple approach often results in processes which work well for the groups represented at design time, but less well when interfacing with those groups that were not represented (e.g. customers or suppliers). However there are structured methods for designing and modelling[1] business processes which are discussed in this section. Process design approaches are split into two broad areas, the clean-sheet approach and the re-engineering/re-design approach.

Clean-sheet approaches to process design are those which do not reference any existing process designs, rather they start with an entirely new business process which is constructed without any legacy or technical debt from existing processes. The main advantage of a clean-sheet approach is that it does not perpetuate assumptions or inaccuracies from any existing infrastructure. Clean-sheet approaches are typically used where there is no existing process to modify, for example when a new product is launched to market.

[1]While designing and modelling business processes intuitively seem different things, the literature in this area uses the two terms interchangeably.

The alternative school is that of process re-engineering or process re-design. Re-engineering approaches deal with incremental changes to a process design which adjust the process to meet some new requirement. These approaches are typically used where a business process has been in place for some time and it does not require wholesale redefinition, for example where a new requirement to record the call handling time is added to a call handling process. In general, process re-engineering is the less risky approach as it does not wipe away the institutional memory inherent in an existing process design, there is always a risk that the assumptions used in a clean-sheet design will not match reality. False assumptions incorporated into an entire process design in the clean-sheet approach are likely to be more damaging than those which only impact a sub-set of the process as would be the case in process re-engineering.

Re-design approaches such as [24] favour an evolutionary approach where the existing process is taken as a base to work from. Aldowaisan and Gaafar's approach requires as input the existing process design and the desired goals of the design exercise (e.g. Reduction in workforce, execution targets, desired capacity). Techniques for process design that work from an existing model are likely to be more palatable to businesses since the risk of negative consequences is reduced over the clean-sheet approach. The proposed approach makes many assumptions which may limit the success of the resulting design when it is used in execution. First, the method assumes that the existing process model is correct and available however the absence of such accurate existing designs is one of the main reasons that the field of process mining (see Section 2.1.5) is developing as an active research area. Second, the approach does not address the environment or so called 'normal accidents' [25] which means the design is likely to be brittle in practice. Personal experience indicates that many process performance issues are caused by the process instances that encounter exceptional or unforeseen issues during execution.

Clean sheet approaches are favoured by Hammer and Champy and such an approach is described in [26]. In order to produce a process design using the Reijers *et al.* approach, a specification of the 'product' to be produced is required. The specification is then broken down into data elements and the set of dependencies between these elements, in the form of a product/data model. The product/data model is used to derive a cost-optimal plan giving the cheapest set of elements which must be calculated for a specified output. The workflow to execute is derived from the set of elements to be calculated from the cost-optimal plan and the dependencies between then (which are known as production rules). Each production rule has an associated cost, so the space of possible sets of production rules can be searched to find the minimal cost of the workflow. Finally the proposed design is analysed for expected performance and validated with the end users of the process. The use of process modelling tools for the analysis of the proposed design is encouraged in this approach and a tool known as ExSpecT [27] is specifically mentioned.

Another technique is proposed by Mendling and Simon [28] which seeks to use the technique of view integration from database schema design and apply it to the business process domain. The suggested design method is to produce, through interviewing process participants or domain experts, event-driven process chains (EPCs) which describe the view each of the participants should have on the overall business process. Once the individual models are available then they can be merged using the method described. This is held to be less error-prone and more time efficient than manually integrating the EPCs. This approach is particularly interesting since it directly addresses merging business processes for different organisations (either internally or externally to the business). Many process improvement programmes that the author has observed have concentrated on the issues caused at the interface of business units, an automated approach that could produce linked processes for the entire super-set of contributors could be extremely valuable.

Once a business process has been designed for a particular scenario it can be deployed into the business. Deployment of a business process might utilise a specific business process management system (BPMS) (for example, one of [10], [11], [29]) which takes the business process design and automatically produces an appropriate workflow. An alternative method of deployment, particularly for legacy processes, is to utilise an existing workflow management system. Using an workflow management system requires an additional workflow design and the appropriate operational support systems to be modified or developed.

While the business process is in-life the support systems produced or modified in deployment are used to support execution of the business process instances in-situ. Systems to control business processes are described further in Section 2.1.3.

This thesis is more closely aligned to re-engineering style approaches since changes are made using the existing process as a base. However no new process design document or process definition is produced and each process execution is dealt with on an individual basis. Therefore the healing of individual business process executions should not be considered to be a process design methodology.

2.1.2 Process Improvement

When deploying a new process, it is unlikely that process instances will execute either as designed or as desired due unforeseen circumstances, knowledge gaps, and incorrect assumptions during the design process. Therefore process improvement is often more important than process design. In the same vein as structured approaches for process design, there are many methodologies for process improvement including the Toyota Production System [8] and Six Sigma [7]. These are discussed further below.

Production management is one approach to the operation and improvement of business processes. The business process and the resources that are required to execute that process are treated as though they are a production line. One widely used production management approach is the Toyota Production System [8] which defines a way to organise the business (and therefore the business processes) to minimise waste and other undesirable properties in order to streamline production and continuously improve. Another approach, related to the Toyota Production System, is Lean Manufacturing [30] which is less focused upon the manufacturing sector. The overall goal of the Lean methodology is to remove resources and tasks from the process that do not directly contribute to the end customer. Both the Toyota Manufacturing System and Lean Manufacturing are intended for use in industries with tangible products, however these approaches have been successfully used in service based industries. Case studies of the application of lean methods to service industries include [31] which shows the main challenge to implementation being one of mindset with the service industry managers not wanting to be classified as manufacturers. A framework addressing some of the potential issues with applying lean methods to service industries is proposed by Ahlstrom [32].

Another widely used approach for process improvement is Six Sigma which was originally developed by Motorola [7]. Six Sigma seeks to identify and remove variation in business processes through the use of trained process improvement specialists, and statistical performance measures. Six Sigma is a data driven approach which concentrates on eliminating defects from the process under control. A defect is defined in Six Sigma as a result that deviates from customer specifications (eg. missing a delivery date, or providing the wrong product). Metrics are defined such that defects can be detected from data recorded during the process execution. Six Sigma aims to drive the rate of failure below 3.4 per million executions (which corresponds to the definition of 6-sigma in statistical modelling).

As a strong emphasis is placed upon measures in Six Sigma it is vital that such measures accurately reflect the business performance that is being investigated. If overly simplistic measures are used them they are open for abuse by staff seeking to inflate their own performance. One disadvantage of Six Sigma, and of metric based approaches in general is that they can lead to process participants disregarding the process to a greater or lesser degree as they try and force the metrics.

A key approach to improving existing business processes is to monitor their execution for anomalies, often this is using a business intelligence (BI) package (e.g. Oracle Business Intelligence [12] or Pentaho [33]). A business intelligence package allows users to view and analyse performance metrics which may include metrics on overall business performance as well as process specific measures. These BI packages are not aware of the business processes being measured and can only report on information that is pre-computed and stored for them to display. The key issue with this and similar approaches is that the decision on what to change rests with a human actor who must discover the problem, diagnose the cause, decide what to do, plan how and when to do it. All but the most trivial changes would require a change programme to be setup leading to significant delay before the change is enacted and significant cost.

The aforementioned approaches all involve significant work by experienced process improvement practitioners, and also require a significant investment in time and effort to gather data on process performance, to discover process issues, decide on an improvement plan, implement the improvements and monitor for the desired effect. Azvine *et al.* [34] suggest using intelligent software for problem identification and to feed back into process execution. The tuning of process execution parameters automatically would allow some level of autonomy in the business process by allowing the execution to react to changes in the environment. However the approach proposed by Azvine *et al.* [34] does not consider structural or organisational changes to the process, instead it considers only parametric changes to the business process (e.g. allowable travel time for an engineer to travel to a fault). Such limitations leave a large space of possible changes outside the scope of the automated improvement mechanism.

There are two major issues which focus solely upon the metric based process improvement options that were discussed in this section. The first is the manipulation of process to artificially enhance the performance measures for the process. For example, if there is a target for Task A to be complete within 1 hour, the team responsible for that task may close and immediately reopen that Task each hour to keep the average time below the target.

The second issue is that knowledge about the process and systems is assumed to be accurate and up to date, and often more problematically, that the process that is being executed is the same as the process that was designed and is documented. A further disadvantage of these process improvement techniques is the need for a human to identify issues, decide upon corrective action and implement the change programme. Understanding the entirety of large scale business processes is not a practical undertaking. Such a process would likely have hundreds of tasks each with its own pre- and post-conditions, its own internal actions, and its own transition behaviour.

2.1.3 Business Process Management (BPM) Systems

Business processes are concerned with the orchestration of tasks in order for the business to achieve a goal (see Definition 1). In the current environment is it likely that many of the tasks in a business process are either fully automated or supported in some way by an IT system, which is either a commercial off the shelf system (COTS) or a bespoke system built specifically for the task. Given the abundance of systems that are acquired or built to support business processes a major challenge is to orchestrate these systems to efficiently support the process instance executions.

Initial attempts at supporting and automating business processes revolved around Workflow Management Systems (WFMS) which are designed to facilitate the flow of work for the tasks of a business process instance execution to the different process participants. WFMSs control the current location and resource assignments for each process instance. WFMSs may also support automated actions but will not usually enable a user to perform the actions prescribed by the business process [21, p. 9]. Examples of systems of this type include bug trackers (such as Bugzilla [35]) which allow users to assign work to each other, and system administrators to configure the required stages of the workflow. For example, in Bugzilla the progress of a bug would be implemented as the status of a bug which might flow as follows: Created $\rightarrow$ Triage $\rightarrow$ Assigned $\rightarrow$ Testing $\rightarrow$ Closed. In general, WFMSs are able to help organise the work, but do not play an active part in that work itself.

van der Aalst *et al.* [36] give an overview of the historical development of systems and IT from workflow systems to Business Process Management Systems (BPMS). Building upon WFMSs, a BPMS should orchestrate the complete execution of a business process from initiation to completion including both manual and automated tasks. In addition to facilities provided by WFMSs BPM systems typically also provide extra facilities in order to support the entire business process instead of just the workflow. BPM systems may allow users to design and deploy user interfaces for manual tasks, to integrate with existing systems (usually via web services), and to monitor the status of current process instances.

In practical terms a BPMS is a system in which a business process can be designed (typically using the Business Process Execution Language (BPEL) [37]) by a knowledgeable person, and then deployed to a component of the BPMS which will manage the execution of the process instances according to the design. Examples of BPM systems that are currently available are JBoss jBPM [10], Oracle SOA Suite [29] and Progress Savvion [11].

There are a number of advantages to using these tools including faster time to respond to process design issues, no requirement for the IT infrastructure teams to be involved in business decisions and increased reuse of process components [38].

Perhaps the greatest disadvantage of WFMS and BPMS approaches is the assumption of a steady-state, or very slowly changing, context. In a modern commercial setting businesses need to be able to adapt quickly to changing market conditions (e.g. a competitor bringing out a new product, or a new competitor). In addition, as businesses become more complex and interconnected a relatively small issue either internally, or in a supplier or customer relationship can have wide ranging effects across a company. The ability of a company to respond quickly and change processes to include workarounds is essential.

2.1.4 Designing Business Processes

There are many representations of business processes that can be used in designing processes and for archiving historical process designs. Many proprietary tools such as *ARIS Business Architect & Designer* [39] have their own representation, or might use the standardised Business Process Modelling Notation (BPMN) [40]. Using a vendor specific notation has the disadvantage of limiting portability of the designs between tools.

Alternatively BPMSs such as Oracle SOA Suite [29] or JBoss jBPM [10] are using the BPEL Standard [37] as their notation with appropriate tooling support. The advantages of using BPEL for design is that it can be executed on workflow engines from a choice of vendors since BPEL is widely supported. The major components in BPEL are presented here, but a full list is available in the specification [37]:

- Partner Links - These reference external web services that are to be used within the process (e.g. a payment processing service).
- Variables store data on the process state, or web service messages.
- Sequences are sequential orderings of tasks.
- A receive is a point where the process to waits for an input.
- An invoke is a call to a service defined by a partner link.
- BPEL uses components such as a switch to provide control structure.

The list of main BPEL components clearly shows BPEL's origins in the Service Oriented Architecture (SOA) environment as they make no reference to the human actors which are an important part of business processes. Human tasks are added in an implementation dependent way by each BPMS. For example, in the Oracle SOA stack [29] human tasks are added by first creating appropriate user interface elements on an application server which can be scheduled and queued by calling a web service from the BPEL execution, in this way the BPEL engine can be used unaltered from a pure SOA stack.

2.1.5 Process Mining

There are many reasons why processes might deviate from their designed form such as local laws and customs, workarounds for process problems, out of date documentation, staff training issues, and so on [1]. Process mining provides an approach to remove or reduce these problems as it provides a mechanism to discover the process that was executed from execution logs retained by the business. Deviations from the documented process and coordination issues can be discovered.

Process mining was introduced by Weijters and van der Aalst in [41] and since that time there have been many algorithms and analysis options proposed. A discussion of process mining algorithms is out of the scope of this document however Tiwari *et al.* have a comprehensive survey [42].

Process mining and process analysis tools such as ProM [43] and Aperture (Section 2.1.5) utilise a simpler representation of business processes than that of BPMN or BPEL. In the process mining representation each vertex of the diagram represents a task or activity within the process, and the edges of the diagram represent transitions between these activities that have been derived in the mining phase. In this fashion a *task* can in fact be any representation of an activity from any of the conceptual levels (see Section 2.1) be that the name of the work being undertaken, an individual, an organisational unit or a high level concept. As an example of the flexibility of this approach van der Aalst *et al.* use the ProM tool [43] to perform analysis on the social network of a large organisation in the Netherlands [44].

Process mining provides a simpler representation than that typically used in process design such as BPEL [37], Aris [39] or BPMN [40]. For example, Figure 2.1 shows a BPMN process that contains a decision point that is used to route the executing processes through the workflow. Process mining of instances of that process using the Aperture Process Mining Tool would result in output like Figure 2.2. Current process mining tools would not be able to derive the condition behind the decision, so a simple fork in the process is shown instead. Decision Mining, is introduced by Rozinat and van der Aalst in [45] and is still an active area of research with recent papers such as [46] discussing the simplification of process models via the analysis of decision points within the process. Other examples of recent work in decision mining include [47] which examines the problem of determining scheduling rules from process models, and [48] which investigates an approach to mine decisions that are not clean combinations of the XOR and AND operators.

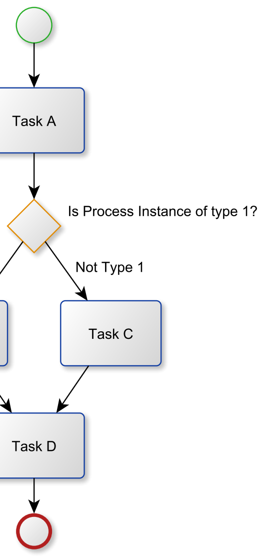

Figure 2.1: A simple process using notation from BPMN 1.2.

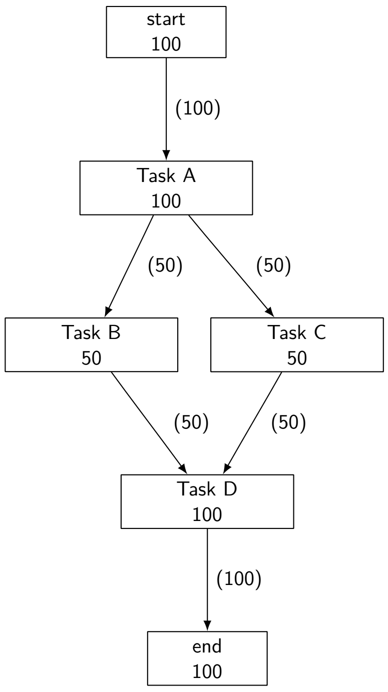

Figure 2.2: Result of process mining 100 process instances of the business process in Figure 2.1 with the Aperture Process Mining Tool. Numerical values are execution frequencies for each task and transition. 50 of the process instances were of Type 1 and 50 of Type 2.

Industrial use of Aperture Process Mining Tool in BT

The Aperture Process Mining Tool is developed at BT by the author, with the intention of realising some of the benefits of process mining and process based analysis in BT. Aperture was initially developed prior to the start of the EngD however it was been extended to allow the extraction of process data for classification purposes (as in Chapters 3 and 4). Aperture will continue to be used as a platform that will enable BT's research in the area of Process Analytics in the future.

Other process mining tools are available. The most commonly used and cited in literature is ProM [43] which forms a central part of academic process mining. It's primary purpose though is the development and refinement of new algorithms, and as such the tool is very complex and difficult for non-specialists to understand. In contrast, Aperture has been designed from the outset to be used by business improvement and process improvement specialists who are not well versed in process mining theory.

In addition to ProM there are now a number of other commercially available process mining tools that are similar to Aperture in that they are tools for business users rather than academic investigations. The most active of these in the process mining community is Disco [49] the authors of which host a process mining mini-conference each year in The Netherlands in conjunction with Wil van der Aalst, the major figure in the academic process mining community. Given that BT owns and can develop Aperture precisely for its own needs, and for the needs of this EngD project, it is the clear choice of process mining tool for the EngD.

Aperture uses the principles of process mining to create models of process instances. Once the set of process instances has been modelled a broad spectrum of process analysis options can be used to help business improvement consultants identify areas for process improvement. Comparison of process instances from before and after process changes can be performed to quantify the effect of the change on the business.

Aperture has been, and is being, used in trials with BT business units collecting and analysing data in order to inform the research and development direction for the tool itself, and to deliver process improvements to the company. Aperture has been trialled across 4 of BT's lines of business, across various types of business process (sales processes, troubleshooting processes, service delivery processes). The volumes of process instances in the trials ranges from hundreds up to millions and the numbers of tasks from tens to thousands.

As a result of providing the process mining tool to users responsible for process improvement 2 of the processes analysed have seen a tangible increase in performance. Other trials have identified issues with the process mining tool itself, and more usually with the workflow and data warehousing systems in the company.

Aperture takes information on the business process execution from the WFMS used by the business and uses it to reconstruct each process instance. The algorithm used by Aperture to derive the connections is based upon minimum spanning tree algorithms [50]. First, the tasks that constitute the elements of the process instance are created. Once processing of the individual tasks is complete, the connections between tasks in the process instance need to be determined.

There are two different cases to consider when connecting the tasks in a process instance, sequential process instances and parallel ones. Where a business process performs tasks entirely sequentially with no temporal overlap between one task and the next Aperture is able to determine the connections between tasks simply by ordering the tasks in each process instance by start and end time and then added a connection between each pair.

However, in the alternate case where the process is either known to exhibit parallelism or the existence of parallelism cannot be determined a computationally complex algorithm for connections is used. This algorithm uses an approximation of a minimum spanning tree to connect the process flow of execution. The cost of each edge in the tree is determined by the time difference between the end of a task and the start of the next. Pseudo-code describing the outline of the connection algorithm is given in Algorithm 1.

In some business processes there are cases where the process execution is required to run concurrently in 2 or more distinct branches. For example, fabricating the components of a car could be seen as part of the process for building the car itself. If we were to process mine the building a car process from the execution traces of the component manufacturers we would likely find that the reconstructed process includes links that are not possible in practice: e.g. the completion of the engine cover by the manufacturer and the start of the moulding of the radio's volume dial by an external supplier. In order to ensure the models of the process instances are accurate for large features Aperture allows the for the integration of business knowledge into the process mining process. Utilising business knowledge ensures that the derived process models are guided by structural knowledge for large features.

Since the Aperture tool is data driven the quality of input data is of paramount importance. There are many issues of data quality that can arise such as missing or erroneous timestamps, systems that do not capture an appropriate level of detail in data and so forth. There is more information on Aperture and the difficulties encountered when performing process mining in large multinational enterprises in [1].

Once the actual business process model has been derived from the executing process instances, it can be used to inform business improvement activities. It is also possible

Algorithm 1: Pseudo-code outline description of Aperture's connection algorithm

```
input  : $P$, The process instance whose connections are to be determined
output : The determined set of connections between tasks in $P$
1  $T$ := array of tasks in $P$ ;
2  $E_c$ := Set of candidate edges (initially empty) ;
3  foreach $i$ in $[1, |T|]$ do
4      foreach $j$ in $[1, |T|]$ do
5          if $i \neq j$ and endTime($t_i$) $\leq$ startTime($t_j$) then
6              insert $(t_i, t_j)$ into $E_c$ ;
7          end
8      end
9  end
10 $E_{mst}$ := run kruskals MST algorithm using set of vertices $T$ and edge set $E_c$ ;
   // $E_{mst}$ now contains edges forming a partially connected graph.
11 $T_{unconnected}$ := find tasks which are unconnected to start and end in $E_{mst}$ ;
12 foreach $t_u$ in $T_{unconnected}$ do
13     foreach $t_n$ in $T$ starting after endTime($t_u$) do
14         if $t_n$ and $t_u$ are not currently connected by edges in $E_{mst}$ then
15             insert $(t_u, t_n)$ into $E_{mst}$ ;
16             check parent of $t_u$ to remove now redundant connections ;
17             break ;
18         end
19     end
20 end
21 return $E_{mst}$ as the set of determined connections for $P$
```

to use the derived model as the basis for a simulation of the process which can be used to examine different options for resolving issues. This model provides an excellent counterpoint to the purely metric driven approach described in Section 2.1.2, which can be mislead by manipulative behaviour of the humans involved in the process. Process mining of these cases would reveal the abnormal execution paths of the manipulated orders allowing the underlying issue to be targeted for improvement.

Figures 2.3 and 2.4 show output from the Aperture tool, in an analysis of two actual business processes at BT. The diagrammatic output illustrates some of the features of Aperture analysis. The numeric values show execution frequencies of each of the tasks and transitions in the business process, allowing the identification of frequent tasks and frequent paths. The appearance of the edges in the figures is used to highlight areas where there is potential inefficiency; solid edges represent transitions to a state that was not previously observed in this execution, and the dashed edges represent transitions to a state that was previously observed, therefore the dashed edges show where task repetition is occurring.

Task repetition can be an indicator of either design or implementation issues within a business process. To take a simplistic view, each task required for the process to complete should need to be accomplished only once; if something needs to be executed multiple times then it is taking away capacity that could have been used to progress more process instances. Task repetition is not always an indicator of a process issue since there might be a need for designed repetition. For example, if a customer reports a fault with a telephony service they call the provider who tests the line and then arranges for a technician to visit. Prior to the technician visiting it would be logical for the provider to retest the line before the technician actually visits the property so that his time is not wasted visiting a fault which is no longer in evidence.

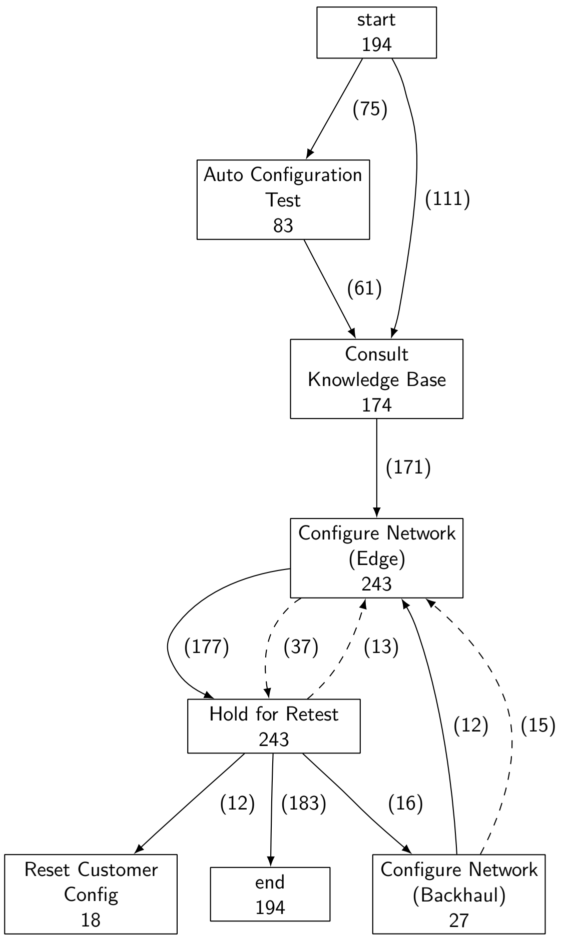

Figure 2.3: Example output from the Aperture Process Mining Tool using data from a fictional process.

Each node represents a task in the business process and each edge a transition between two tasks. The numerical labels for both nodes and edges are their frequency of occurrence. Solid edges indicate the the target of the edge is the first occurrence of the task in the process execution, dashed edges indicate that the target task has been previously executed (therefore indicating where repeat work is being performed in the process).

The diagram has also been pruned to delete edges that occur less than 10 times to remove noise from the diagram and aid comprehension of the main process behaviour.

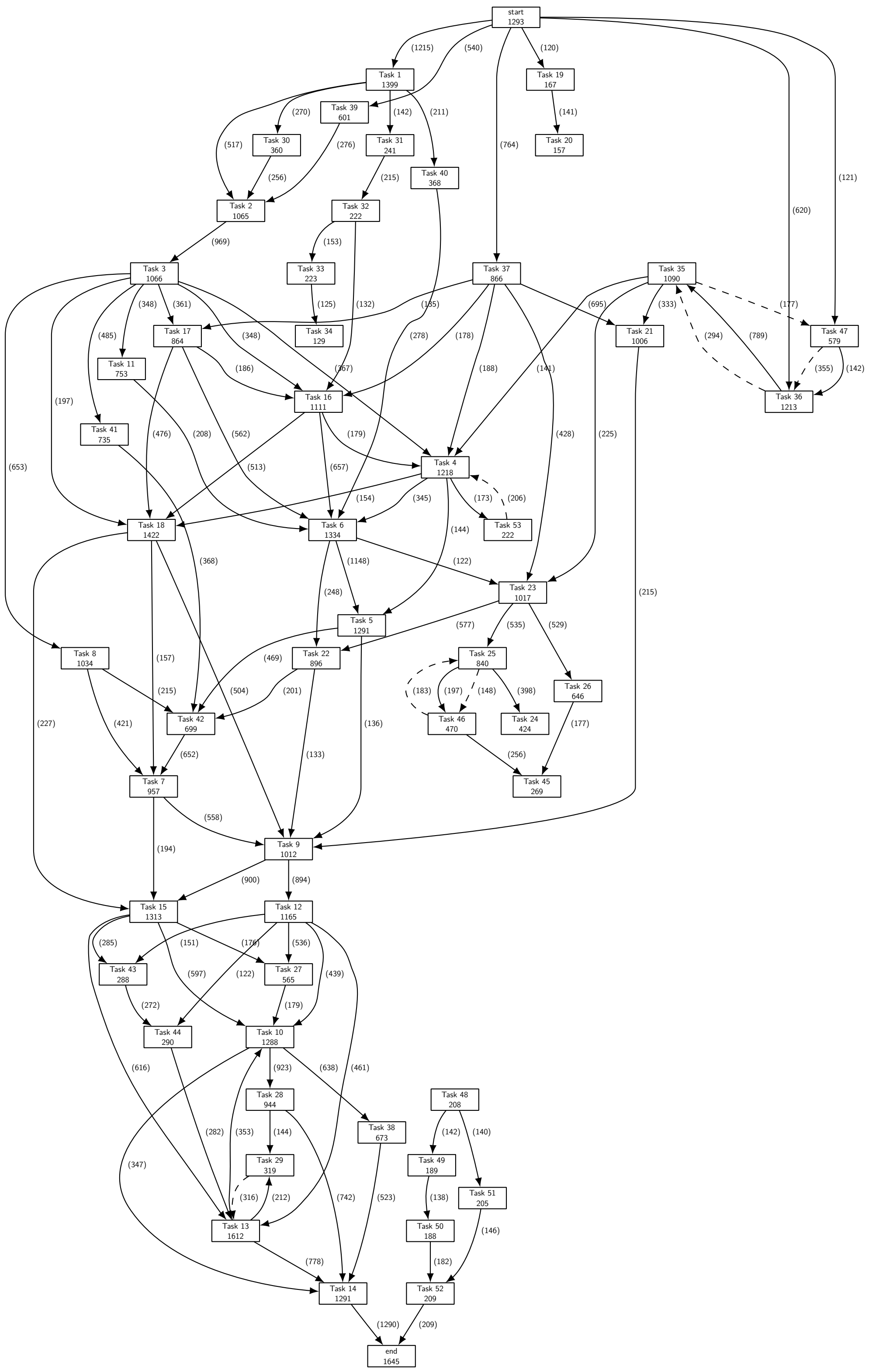

Figure 2.4: A model of a business process that provides new resilient data services to multinational customers. The process involves staff in the countries at either end of the circuit and in the UK. The sample includes for a 1293 process instances, and shows only edges that arise in more than 90 per cent of process instance executions.

The numerical labels show frequency of occurrence. Solid edges indicate the the target of the edge is the first occurrence of the task in the process execution, dashed edges indicate additional executions.

2.2 Autonomic Systems

In this section the different self-* properties are discussed and their possible effects on the world of business processes is highlighted. From experience interacting with people responsible for business processes it seems that a purely autonomic process with no centralised control is unlikely to be adopted because the business would see the lack of a central controller to mean there is no process control; therefore in addition to true autonomic approaches some centralised approaches are also discussed which provide some of the self-* properties while sacrificing some robustness for centralised control.

2.2.1 Self-Configuration and Self-Organisation

A self-configuring system is one which is able to adjust its operations based upon changes in its operating environment [23]. These could be changes in the components of the tasks, changes in available resources or changes in the components of the system itself. This property has clear links into the management of IT systems and IT estates where hardware and software are involved in a perpetual refresh cycle, and the user community and uses of the systems are constantly changing [51].

A principle that is related to Self-Configuration is that of Self-Organisation. In this context Self-organisation is the automatic arrangement of the components of a system such that the system is able to complete some task, or meet some goal [52]. In this document the term self-configuration is used as an umbrella term for both self-configuration and self-organisation.

What constitutes self-configuration varies with the type of system that is being configured. In an IT system the configuration might be which network links should be used, or which servers should run which applications; in a robotic production line it might be which robot executes its task in which order. The configuration (design) of a business process incorporates the execution order and hand-offs within the process flow and the allocation of tasks to actors within the system (which could be human or robotic - see Section 1.1). Business processes are therefore most similar to systems with independent actors (e.g. multi-agent systems [53]), or compositional systems (e.g. web services [54]), and are less similar to systems such as computation clusters, data centres, and networks.

Techniques for self-configuration are generally reliant upon concrete information about the actions of the components of the system being available. This could be an OWL-S [55] specification or a web service description (i.e. WSDL [56]), a list of goals and preconditions in multi-agent and planning-based systems [57] or some other representation. Concrete information allows tools to reason about the world and the available resources. Producing an appropriately rigorous description for a business process is a non-trivial task because of the cost of discovering, validating and maintaining this information across hundreds (if not thousands) of processes and many more constituent services. A further difficulty is the inclusion of human actors within business processes which often leads to significant variability in processes which would make predicting the progress of the process difficult [1].

An additional challenge in self-organised systems is matching the organisations that are generated by self-configuration techniques and the goals of the business. For example, each agent might have a cost implication and the goals to minimise cost of the solution and to achieve the highest throughput of tasks could be in conflict. In business processes there are usually several competing goals, and their relative importance can change as in field of multi-criteria optimisation [58].

Decentralised Organisation

The essence of self-configuring/self-organising systems is their ability to organise without a requirement for a central controller using only location limited communications and sensing abilities [59]. Decentralised systems have the advantage that they are not vulnerable to the failure of the central controller and as such can be made more robust. However the conservative nature of business regarding process management means that it seems unlikely that businesses would adopt a decentralised approach. Experience indicates that it is often felt that a slow rigid process is seen as preferable to a fast but variable one. One rejected research direction for this work could be to investigate how a self-organising business process could be constructed from high-level components which are themselves predefined and centrally controlled.

Swarm based approaches to decentralisation place a number of individual elements into a shared environment. Each of the elements within a swarm has only a limited ability to sense the environment and to communicate with other elements and only has a fixed number of actions that can be taken [59]. In spite of the limitations of each element within the swarm the emergent behaviour can be very powerful and very robust to environmental change, particularly in natural systems e.g task allocation in ant colonies [60].

One issue with decentralised approaches is ensuring that the rules of the individuals in the swarm are appropriate for the swarm to achieve the overall goal. This is particularly important because business processes are not effective or complete if they do not complete their work. Winfield *et al.* propose the discipline of swarm engineering [61] to bring dependability to swarms. The proposed approach is an iterative one with 3 phases: analysis, design and test. The analysis phase determines if the system is able to perform desired actions, and that it does not exhibit undesirable behaviours. The design phase suggested is an extension of the Yourdon structured design methodology. The test phase uses statistical methods to measure the times in which the swarm performs the desired behaviours. The paper focuses on safety critical systems which would typically have more stringent dependability requirements than those of business processes, however the points made should be applicable even in the weakened business process case.

Other decentralised organisation approaches can be seen in swarm robotics. Swarm robotics is a field concerned with enabling a group of robots to co-operate to achieve a particular task or function in a given environment [62]. This is conceptually similar to the way a business uses groups of people to perform a business process.

One such approach is an evolutionary robotics one proposed in [63]. Evolutionary Robotics is a bottom up approach where variations are made to the each robots "genome" and then the selection for breeding is determined based on the performance of the overall swarm. Tuci *et al.* 's approach evolves neural network based controllers for each robot that allow them to self-organise at runtime to perform their task.

In [64] it is argued that some of the research on swarm robotics can be helpful in an immune-inspired self-organisation in complex computer architectures, citing [65] as an example of such an approach. An alternative immune-inspired approach is proposed by Capodieci *et al.* [66]. In both cases at a high level each robot contains a lymph node that contains a set of antibodies describing when different modes of operation apply. The robots are able to trade antibodies with each other when in communications range which allows the improving co-ordination rules to be spread throughout the swarm. As with many swarm robotics tasks this case study is to perform a foraging task that is much simpler than a business process would be.

Business processes often contain time-constraints. For example, that a field engineer must visit the property for an installation at the agreed time. The approaches to swarm robotics discussed thus far in this section are not able to deal with this problem. Khaluf and Rammig [67] describe an approach which allows a swarm of robots to allocate tasks between themselves whilst respecting the deadlines for each task without the requirement for any central control. Any work on self-organising business processes would also have to deal with these issues.

Another decentralised organisation approach is that proposed by Kota *et al.* [68]. The paper describes how a system of agents would be able to modify their relationships in order to ensure efficient task distribution and allocation throughout the swarm. The performance of the system is shown to be approaching that of a centralised allocator [68] which should provide evidence to businesses that decentralised solutions should not be written off on the basis that the trade-off between efficiency and resilience is unfavourable.

Self-Configuration in Multi-Agent Systems (MAS)

MAS are those systems which consist of multiple independent autonomous agents that are able to self-configure without the requirement for central control [69]. The systems are made up of agents which coordinate actions between themselves, allocate resources and so on. Since such systems evolve together with their environment and adapt as it changes; they are robust to changes in that environment [70].

MAS agents have a number of important properties which make them useful in this context [53]:

- Autonomy - Agents should be able to work alone and without external intervention, and they should have some control of their own state.
- Social Ability - Agents should be able to communicate with each other.
- Reactivity - Agents should detect and respond to changes in the environment.
- Pro-Activeness - Agents should take actions to move the system towards its goal even in the absence of changes in the environment.

One approach to self-organisation in such a multi-agent system is Unity described by Tesauro *et al.* [71]. Unity uses an approach called "goal-driven self-assembly". Each agent starts with a very simple description of what it is supposed to do. The agents register themselves in a registry and then use that registry to determine the agents that can provide facilities they require for execution. Ultimately the agents are connected via the negotiated relationships to provide services to each other and the external environment. This approach allows agents to be added and removed dynamically with re-configuration taking place as appropriate.

A type of multi-agent system is known as an holonic multi-agent system [72] which extends a standard MAS by allowing the participating agents to form larger super-agents. These holonic MAS can also support self-organisation with a suitable approach such as [73]. In this way agents providing low level functionality may decide to work together to present a larger block of functionality to the rest of the system. This is analogous to the way that a business might organise itself in to teams of staff that work together to accomplish the team's and the organisation's goals. In actively utilising the larger structures, the holonic approach would appear to be a good fit for a partially autonomic system where the larger structures might be created manually and then allowed to self-configure at the level higher than that of individual agents.

Another related approach is illustrated by Itao *et al.* in [74]. Jack-in-the-Net provides a mechanism by which Cyber-Entities (similar to agents) with simple behaviours are able to dynamically self-organise into Super-Entities in order to provide some service to the world. This idea could be very useful in the business process context, if for example, the systems that were necessary for a business goal were able to amalgamate and present a single 'process' to the world that could be invoked e.g. a customer email service, a line test application, and a fault diagnostic library could combine to form a complete fault resolution service. One area that Jack-in-the-Net does not address is that of prioritisation or queuing of incoming work which is essential in business processes.

Multi-agent approaches such as these have the advantage that they are able to adapt to the environment and provide a system to perform a task given the appropriate goals and agents. The main deficiency in these techniques in terms of using them for business processes is that they do not allow human actors to take part in the system. In addition MASs of autonomic agents may not be able to replicate some constructs that are part of business processes, for example if a service test requires a technician to be present at each end of a circuit then the dispatcher agent would need to recognise and schedule this appropriately which appears to be non-trivial for autonomic agents. Self-configuration in business processes could incorporate different tasks at different levels of complexity. It could be that the configuration simply decides upon task allocations to different applications and human actors, alternatively the configuration might be the temporal organisation of tasks within the entire business process.

MAS and Artificial Immune Systems

There is also a body of work in which Multi-Agent Systems are constructed using approaches from Artificial Immune Systems (AIS). One example of such an approach is described by Dasgupta [75]. Dasgupta proposes a framework that uses an artificial immune inspired approach to create a multi-agent decision support systems which have the desirable properties of MAS that are discussed above.

A further AIS inspired approach to MAS is described by Sathyanath and Sahin [76]. In this paper Sathyanath and Sahin propose a generic model called AISIMAM for building self-organising multi-agent systems. They demonstrate the technique using a simulated mine clearing exercise where multiple independent robots are co-operating. This has some similarities to the business process domain where multiple teams can be co-operating on a single process, however the proposed system contains only a single type of agent which is not generally the case for business processes.

There are however artificial immune inspired systems which are not multi-agent systems as they do not exhibit all of the properties of multi-agent systems discussed at the start of section 2.2.1. For example, classification algorithms constructed using the principles of AIS (eg. [77] and [78]) can be thought to exhibit autonomy (of agents) as the recognition cells in the classifier function independently, but they do not communicate with each other as co-operative agents would.

2.2.2 Guided Self-Organisation

A system requiring central control is not one in which the actors are operating autonomically because each actor is not managing itself in response to polices from the administrator [23]. In spite of this such systems might be desirable for real businesses because central/hierarchical control is a more familiar organisational paradigm than a fully distributed, flat organisation. It is possible that some of the benefits of a self-configuring system could be realised without the system being truly autonomous. In this section approaches that require global knowledge, or central control are introduced.

Planning Inspired Approaches

One area of research which has applicability to self-configuration is planning. Given a goal that is described in some way, and a set of actions that are available for execution, a planning algorithm is able to a determine the series of tasks required for the goal to be accomplished. Algorithms for planning are numerous and varied, some examples are STRIPS [79], GraphPlan [57] and SHOP [80]. A disadvantage of these approaches is that they rely on knowledge of the overall goal of the system and on the system's current state so that actions can be appropriately planned by a central planner. A truly autonomic system would not require a central controller or planner.

A rudimentary approach to self-configuration using planning is used in Kreno [81]. Kreno utilises the GraphPlan algorithm [57] to plan the execution of the appropriate tasks based on the user requirements. The tasks available to the planner are discovered from a standard Universal Description, Discovery and Integration (UDDI) repository and the domain knowledge required to produce the plan is generated from OWL ontologies. Based on the available resources Kreno creates a description of a planning problem which it then passes to GraphPlan for a WS-BPEL plan to be produced. The final stage is the execution of that plan on a standard BPEL execution engine. In its current state Kreno is also limited to influencing new process instances at their inception so environmental changes detected during the execution of a process would not be able to influence the execution.

Kreno's approach includes an element of self-configuration, in that it produces plans without supervision and allows them to be executed. Such an approach could be extended to allow the OWL corpus of domain knowledge to be updated with feedback from executions of previous plans, allowing the system to respond to environmental changes in an autonomic fashion (this is similar to the case-based reasoning approaches to self-configuration, see Section 2.2.2).

One weakness of planning-based approaches such as [81] can be in the incorporation of contingency or runtime information into the plans. That is to say that a plan which seems sensible at 09:00 may be infeasible at 10:00 if one of the agents is taken offline due to a fire, for example. It is essential that self-configured business processes are able to deal with interruptions. A naive approach would be to re-plan after each step but this would introduce considerable overhead.

Lundh *et al.* [82] describe a system which aims to resolve some of the issues with changing environments. The system also takes inspiration from planning, and is able to produce a configuration of robots. The configuration is then monitored to detect changes in available resources, and reconfigured to accommodate the changes. Reconfiguring the system is performed when an environmental monitor detects some change, e.g. a system going offline. However, this introduces the issue of determining what failures should be monitored which can either be done manually, or perhaps could utilise some of the change detection approaches discussed in Section 2.2.5.

Case-Based Reasoning

Case-Based Reasoning (CBR) [83] uses a database of historical information to make decisions about the best actions to take in a new situation. The large amount of historical and archived data on process executions together with their metrics mean a large database of previous cases can be derived. Approaches such as [84] provide mechanisms to use CBR in self-configuration scenarios.

A potential weakness of a CBR solution is that being guided by the past is only useful if there is only slow evolution in the environment. In the case where there is rapid environmental change, a CBR approach may suggest a next task that has previously been reliable but could now be undesirable due to changes in costs, unreliability issues, or other problems. A more practical disadvantage of CBR approaches is the amount of storage space required to hold to corpus of prior executions.

The advantage of CBR approaches over other approaches in this section is that they do not require a specific expert to create descriptions of the components of the system but they are able to learn from the actions of previous executions. This is desirable in a business environment where the number of tasks to be annotated may number in the thousands.

Web Services Composition

Businesses are increasingly using SOA and BPMS to expose and compose business processes from existing functionality. These separate pieces of business functionality are usually exposed to the BPMS/SOA as web services. Therefore web service composition provides a possible basis for the implementation of a self-configuring system. Compositional approaches were studied as part of a general review of self-* systems whilst the focus area for the thesis was still being decided. Self-configuration of business processes was not pursued because the implementation of such a system would be of significantly higher risk than augmenting an existing process with self-healing elements. Self-configuration is more risky because the business would not be able to perform the process at all if the self-configuration system failed, whereas it can continue to serve customers with the existing process even if the self-healing parts are non-operative, albeit with a higher failure rate. The compositional approaches discussed in this section do not directly impact the self-healing focus of this thesis, nor the proposed solution.

In a web service-composition approach, semantic information is used to annotate services. A composition algorithm then reasons over the annotated elements [54]. This is not strictly a self-organisation approach itself because the configuration is considering only the semantics of the available services and is not considering the execution environment. If each of the tasks within a business process could be exposed as a web service, then an appropriate composition of those web services could execute that business process. As with Kreno [81], the requirement for a central planner or composer prevents this being a true self-organising approach.

Planning approaches such as the GraphPlan [57] approach used by Kreno [81] are not able to reason over predicates that might be applicable only for a certain time window. In business processes the requirement to be active in a certain window is important, for example, a technician may only be able to gain access to a customers premises for a certain time, or a resource reservation might only be applicable for a matter of minutes. However approaches such as [85] could replace GraphPlan in the Kreno system to allow such reasoning to occur.

One such Semantic Web approach to combining web services that might also be of use is described by Traverso and Pistore in [86]. Models of each service should be provided in OWL-S [55]. The OWL-S models can be translated into automata, as can the requirements for the service to be built. Given the automata for the source services and the target, [86] uses the “Planning as Model Checking” [87] approach to plan the composite service. The generated configuration can then be translated into BPEL4WS [37] for execution on standard infrastructure.

Another approach to service composition is proposed by Yu *et al.* [88]. In this approach

Genetic Programming is used both to select web services to meet the business goal and to optimise the selection of services based on their QoS characteristics. This would give the business confidence that the produced organisation of services would have acceptable performance characteristics. The other approaches discussed in this section would not give such an indication as they do not consider service performance in their selections.

2.2.3 Self-Healing

Self-healing is the ability of a system to continue to operate in spite of the degradation or failure of some of the components of that system [23]. In terms of business processes this could be failure of either the systems supporting automated parts of the process, or 'failure' of the teams executing manual tasks in the process. One difficulty introduced by the split nature of business processes is that is can be difficult to detect if a manual task is completed correctly because those tasks are given to human to exploit their ingenuity, however issues might arise later in the process execution as a result of mistakes. Automated tasks would typically be unresponsive over network links, or return error messages, but human actors may not be so helpful. Example studies of fault-tolerance in financial systems are discussed in [89] and in [64]. Self-healing systems use a detect-diagnose-heal cycle which requires instrumentation of the system being healed to allow for detection, and the ability to reason about the system to determine the cause of issues and the corrective action to be taken [90].

Architectural Approaches

The architectural approach to self-healing provides resilience at the level of the system infrastructure and requires that the system be designed with that system or architecture in mind. In general these systems use an observe-monitor-repair cycle in which the system is observed for its performance characteristics, these are checked to determine if the system is in an acceptable state and if it is not then changes to the architecture are made to repair the fault [91].

One such approach is given by Garlan *et al.* in [91] where a notion of style is introduced to adapt the framework of the specific application that is running, by providing the important performance metrics, available architectural operations, and some repair strategies that are applicable to the operators and architecture. Repairs are performed when a performance issue is detected by applying the repair operators to the architectural model maintained in the system, and then translating those architectural changes via a translator to the runtime system.

A similar approach is presented by Park *et al.* [92]. They describe a system with a monitoring layer, diagnosis & decision layer and an adaptation layer which runs in parallel to the running system rather than as part of it. Within these layers, six processes run monitoring, filtering, translation, analysis and diagnosis & detection. This architecture extends other approaches like [93] which monitor logs and reports coded into the application for execution issues by also monitoring the hardware and operating system for non-logged issues. Remedial action is determined using a rule engine that uses predefined user-specified actions.

The architecture of the system that includes self-healing is important for determining what problems and solutions can be used. For example, [91] and [92] propose their approaches in software systems which have flexible architectures, applying these approaches to a enterprise architecture where politics and vested interests play a part would be a major challenge. In some domains certification of designs might be required which would require the re-certification of the system each time a architectural change was made, in that case architectural approaches would seen infeasible.

Immune Inspired Approaches

One option for self-healing in complex systems is to again take inspiration from the immune system of mammals [94]. Immune systems in nature are able to respond to maintain their organism in the face of internal and external environmental changed by detecting these perturbations, diagnosing the issue and correcting the issue.

Looking at artificial immune systems (AIS) with the area of business processes in mind, an approach detailed by Russ *et al.* [95] describes how an AIS approach could be used to allocate tasks to computational resources in a grid. H-cells in the system are used to monitor resources and to manage the allocation of programs to those resources. S-cells in the system would monitor the executing programs themselves. H-cells which need to learn the systems characteristics throughout its life-cycle do so using the Hierarchical Adaptive Response Theory (hierarchical-ART) classifier. The H-cells learn through the systems life so are able to adapt to changes as the nature of the workload changes, this is essential in a business process environment where change due to both internal and external factors is virtually constant. The design of the S-cells of the AIS in this approach was to be completed at the time of Russ' publication.

An advantage of an AIS approach is that it is highly decentralised which can improve the resistance of the system. In addition the adaptation of individual H-cells enables the system to adapt to localised changes in the overall environment leading to fast and expert changes being made. A weakness of this approach is that different cells would need to be developed to monitor each type of resource and each type of work, in a business process the number of each of these could be very large, in addition is the issue of monitoring human actors each one of which could be considered a different resource.

Guangzhu *et al.* introduce an approach to adapt AIS to multi-agent systems [96]. The IMEAS algorithm is able to use its virtual cell types to efficiently calculate which agent should receive a particular job without requiring a complex and extensive mathematical model of the system to be defined. Details of the large number of virtual cells that would be required are not discussed so it is not clear if such a system is computationally feasible in a large scale system.

Self-Healing in Business Processes

Self-healing is most closely related to dealing with the failure of components or parts of the system with that characteristic (e.g. the financial example used in [89] and [64]). However in a business process the notion of failure of a component is rarely clearly defined. Alternatively the failure of an instance of a business process is generally well defined by the metrics of the organisation. For example, company guidelines might state that faults on a residential phone line should be fixed within 8 hours, designing and commissioning an Ethernet link between 2 countries should take less than 3 months etc. With that in mind self-healing of a business process becomes more about resolving, or avoiding, issues in the execution of the process than about fixing failures of the infrastructure supporting the process. Of course, failure of the infrastructure may imply failure of the process instance but that is not a given.

There is also the question of how a business process should self-heal and in which circumstances. Detecting the failure of a business process is the first issue, if a single process instance fails from the set of 1,000 that are running that day then no healing is likely to be required, however if 500 out of the 1,000 are failing then some remedial action would be required. Diagnosing the system could also be problematic, in some cases components might signal failure but not cause any process instances to fail. Once the problem is diagnosed then some healing is required, the most straightforward method of healing would be to change the business process to re-route the process executions around an alternative path, for example bypassing a task which requires a failed system. This approach is related to self-configuration of the process. Healing a business process at the component level would require the 'tasks' in the process to be re-engineered and implemented with self-healing in mind which is likely to be infeasible in terms of the effort required.

One approach to self-healing which has direct applicability to business processes is that demonstrated by Baresi *et al.* [97], Their approach uses BPEL and business rule engines (specifically from JBoss) which are either already used to drive business processes, or are being examined for adoption in many businesses. The extensions made use pre- and post- conditions specified in WSCoL (a language designed for this system) to generate assertions that can be checked at runtime to discover issues, and recovery strategies that can be executed specified in a language called WSReL. The recovery strategies that are applicable to each failure scenario are designed by the users and entered into the system, they are not however automatically generated or suggested. WSCoL and WSReL are described in [98]. Detection of issues within the process is performed by the WSCoL probes that are added to the business process which check conditions hold at specific points. Diagnosis takes the form of selecting the most appropriate recovery strategy from those applicable to the currently violated constraints based on priorities defined in each strategy. Healing is then performed by executing the selected recovery strategy on the individual process instances that has failed.

2.2.4 Self-Optimisation

Self-optimisation is the process by which a system can monitor and tune its use of resources to best fit the system's overall goals [23]. For example, a fault management system might have a time target for resolution to meet and a staffing level target - when faults are flowing into the business at a higher than normal rate the system could allocate more line-testing applications to prevent a backlog of test requests building up. The self-healing of a system in response to failure outlined above could be considered an optimisation of that system in this way - new resources are acquired to replace faulty ones.

Optimising with Workload Changes

In one type of self-optimising system, the parameters of the system are adjusted automatically to tune the system based on changes in its workload. Dynamic workloads are a feature of many systems particularly those that are involved in interactions with external parties (e.g. the rate of customers calling a customer service line). One method for optimising a system in response to changing workload is by monitoring its level of current performance and then making some changes to the system based on that measurement.

An example of that approach is described by Menascé *et al.* [99]. Their approach uses the notion of Quality of Service (QoS) as represented in a goal-function and a predictive model of the system to enable the controller to regularly sample the system state, perform a search in the space of possible configurations for a prediction of an improved QoS, and then for that controller to issue instructions for the system to change into the selected state, this tuning would need to occur regularly to be effective. The paper describes an extension to the basic proposal of fixed-rate sampling of performance, to allow the sampling rate itself to be varied. Experimental results from the paper show an improved result is generally achieved.

Another approach to optimising QoS measures in a dynamic system is given by Stantchev and Schröpfer [100]. They propose a method which is able to map the business requirements of a process onto the technical requirements of the infrastructure supporting it. Where a performance shortfall is detected their proposed system aims to improve the performance of the system by dynamically starting additional instances of the overloaded services which can then share the load. The efficiency of this approach would depend upon the ability of the process to support parallel execution in an efficient way. In general business processes separate instances of the process are usually independent, so can be efficiently executed in parallel provided sufficient resources are available.

Optimising with Service Changes

In contrast to the approaches described above Gehlert and Heuer [101] describe a situation where some self-optimisation is possible - the addition of a new service/resource to the environment. There are many reasons why this might occur including: service maintenance, the introduction of a new version of an existing service with bug fixes, a new product requires some new functionality which is not present in the initial environment, or new copies of a service deployed for resilience. The paper describes a goal-based approach to this situation where both the new service and the existing system have goal models which can be reasoned with. Upon the introduction of a new service the approach first checks to see if the new service replaces an existing one using the provided goal model, if the service is able to take part in a replacement the goal models are further examined to determine if one or more requirements is improved by the new service and if so a replacement is arranged.

One limitation of this approach is that it applies only to well defined, automated services which are easily started, stopped and replaced. In a business process where many tasks are to be performed by hand it is not clear how an 'improved' version of the service could be either developed or inserted into the environment.

Self-Optimisation of Business Processes

Self-optimisation of business processes sounds like a panacea, potentially providing a process which is able to run optimally without any of the expenditure currently required on business improvement methodologies and practitioners. For processes involving human actors there is the complication that the resource pool for each of these tasks cannot be changed quickly, if they can be changed at all.

Nevertheless the optimisation of the automatic parts of the process may still yield important gains in efficiency and performance depending upon which tasks on the critical path of the process are able to be improved.

2.2.5 Self-Protection

Self-protection is the ability of a system to discover and resist attacks on the system, irrespective of the source [15]. This is a specific definition that avoids self-protection from overlapping with the other self-* properties, primarily self-healing.

As for self-configuration, approaches to including self-protection into a system can be inspired by biology. D'haeseleer *et al.* [102] propose the use of immune-inspired mechanisms to detect change in a distributed system. Two algorithms are given to efficiently generate detectors, which can be done multiple-times independently in a distributed system. The paper claims that this would give higher change detection rates that a single set, in addition to the removal of the single point of failure. Once the detectors are generated they are able to check the strings that are present in the system and determine if they are self-strings (part of the system) or non-self-strings (foreign or malicious). The action to be taken when a change is detected is left open.

The approach of D'haeseleer *et al.* is extended by Sterritt and Hinchey [103] where three elements are introduced into the multi-agent system to facilitate self-protection. The first is the notion of apoptosis, which is where biological cells self-destruct when a continuous signal telling them not to is removed. The second is the notion of cell quiescence, a way of parking an agent in a dormant state so it can be verified as safe before it can rejoin the actual system. The third is a signal, which they call the ALice signal, which is used by the system agents to securely identify each other as having authorisation to be part of the system. These elements are used to allow agents to self-destruct if they cannot be determined to be valid and correct thus protecting the system from the introduction of misbehaving agents.

The main limitation of the approaches proposed in [102] and in [103] is that they are dealing with agents acting either maliciously, or correctly but not both. In business processes, particularly where humans are involved, it might be the case that the agent is acting maliciously only part of the time. It might also be the case that what appears to be malicious behaviour may actually be a mistake or misunderstanding on the agents behalf (for example, where a service is disconnected erroneously because the agent misunderstood the customers instruction). Choosing the point at which mistaken/erroneous behaviour is classified malicious and then agent removed from the system is not straightforward.

Self-Protection in Business Processes

Of all the self-* properties, self-protection is the one which seems to have the smallest applicability to business processes at the industrial partner. Protection of the system from an external attacker is not something that is regularly considered in this context as the operators who have access to the system are part of the business process themselves, and are therefore not external to the system. The elements that are of most concern are addressed by other properties: failure of a process to complete is addressed by self-healing, failure of the process due to workload changes could be dealt with either by self-organising methods or by self-healing.

The most appropriate issue to examine for self-protection seems to be the protection of the system from misbehaving actors within the process itself, these could be compromised computer systems, disenfranchised human actors, or simply mistakes made by either. In other words preventing the system from failure due to malicious interactions.

Self-protection would likely be more appropriate in other contexts outside of the telecommunications processes at the industrial partner where the external environment is hostile and systems need to resist attacks alone. One extreme example of this would be in autonomic space exploration [103]. Another example of an area on interest for self-protection might be in the automotive industry where self-protection is desirable to protect vehicle systems from issues introduced by exposing the systems to external communications [104].

2.2.6 Discussion of Self-* and Business Processes

The applicability of each of the self-* properties to business processes is varied. One of the hardest parts of managing business processes is designing new processes, the use of self-configuration seems an ideal fit for this problem.

Businesses are constantly reviewing and improving their business processes to reduce costs, increase efficiency, and increase profitability. The area of self-optimisation seems ideal for this task, although the review literature does not seem an ideal fit for the business process area.

The issues of self-healing appear to be best dealt with from the perspective of reconfiguration - that is to say that the process could self-heal by redirecting their execution path when a problem occurs. An alternative would be to change each task to support self-healing implicitly (eg. a line test service that repairs itself rather than requiring the process to instead use an alternative testing service). Re-implementing all of the tasks within a business process would require an enormous amount of implementation effort which is very likely to be untenable, therefore approaches that can incorporate existing services with little or no modification are to be preferred.

Self-protection is not a large area of concern for businesses in my experience, those actors executing business processes are usually trusted (whether humans or machines). However it is possible that some of the self-protection principles could be applied to protect against failure of the process execution, but fixing process instances would be closer to self-healing.

The major issue facing the project is how to include human actors in an autonomic system in a way that allows the agents in the system to interact with them appropriately without forcing the human actors to function like production line robots.

2.3 Classification

Classification techniques are designed to address the problem of determining the label of a new observation based upon a set of training data that is already labelled. Classification is a well known area of machine learning [105] that has been extensively studied. Section 2.2 has reviewed areas of autonomic and self-* systems that utilise classification techniques. For example, self-healing often requires the use of a classifier to determine when healing would be required. Chapters 3 and 4 utilise classification to make a prediction on the outcome of a business process execution and Chapter 5 utilises classifiers to determine potential recovery paths. This section will review first the classical classification techniques and then the immune inspired techniques used in the research.

2.3.1 Classical Classification Techniques

The first of the algorithms used in this thesis is J48 which is a Java based implementation of C4.5 [106]. C4.5 is a development of techniques such as ID3 [107] which produce decision trees suitable for classifying new instances from a set of training data. These techniques are attractive to the industrial partner for two reasons. First, they produce human understandable models which means the model itself can be an input to business improvement programmes. Secondly, decision trees are extensively used internally for use cases such as churn prediction [108] and determining the risk involved in reactivating dormant telephony circuits.

The other classical technique used in the thesis is Association Rule Mining [109]. The RIPPER algorithm is able to efficiently process noisy data sets such as business process data. As with the decision tree techniques the rule base is human understandable so can be used as an input to business process improvement projects. There are also existing use cases for association rule techniques as the industrial partner in diverse areas such as network appliance failure prediction and video recommender systems.

Not all misclassifications are equally impacting on the business. It was desirable to utilise a technique that would penalise the classifier more for missing an impending process failure than for mistakenly flagging a success as a failure. One mechanism for adding this cost-sensitivity to classifiers is to adjust the training data using stratification [110]. This means however that we are either duplicating training data or removing potentially useful observations from the training data in order to re-balance the classes.

An alternative to the stratification technique is MetaCost [111]. MetaCost is able to wrap any standard classifier and provide cost sensitivity to the resulting classifications. This gives a standard mechanism that can be used to address both the class skew inherent in the business process data and to ensure the more costly missed failure cases are treated appropriately.

The introduction of MetaCost does however introduce additional parameters to the learning problem in the form of the required cost matrix. One option to tune the cost matrix for MetaCost would be to use human intelligence however this very laborious and time consuming work for a person who could be better employed. If a human is not able to be used for tuning then there are two options: to use alternative classification techniques that may be more amenable to the raw data as in the next section, or use an automated optimisation technique.

One automated optimisation technique that could be used would be Genetic Programming (GP) [112]. GP based techniques use the principles of natural selection to develop solutions to problems over a series of evolutionary steps. For the parameter optimisation case it would be possible to set up a GP to optimise the parameters of the cost matrix for MetaCost based on the performance of the classifier produced in each run. A example cost function that could be used for this would be the potential monetary savings for the business which could be calculated using the performance results of the classifier. This thesis however is concerned with determining the viability of a self-healing business process system for generic business processes and not to optimising the system for a single process, therefore the investigation of alternative classification techniques that may have wider immediate applicability was preferred to investigating the optimal classifier parameters for the single example data set.

2.3.2 Immune Inspired Techniques

Artificial Immune Systems inspired techniques are those which attempt to solve problems with the application of techniques from immunology either literally or as inspiration. de Castro and Timmis [113] introduce the area and cover the relationship between AIS, Computational Immunology and Computer Science/Engineering. One area that AIS are used in is classification. These immune classification techniques can be used in similar ways to the classical ones and perform the same function as the classical classification techniques discussed previously. Since AIS classification techniques are structured so differently to classical ones, it was desirable to test a number of algorithms to determine their performance characteristics on business process data.

Chapter 4 focuses on the testing of a number of immune inspired classification techniques. This section introduces the area and the techniques used with more detailed discussion on the details of the algorithms used discussed in Section 4.2. These techniques are new to the industrial partner and have the potential to be effective in areas wider that just autonomic business processes. These techniques were favoured because they have ready implementations and as such support rapid testing on new data sets. Additionally the chosen techniques cover a number of different approaches from within the field of AIS classification.

AIRS is a family of classification algorithms proposed by Watkins *et al.* [77]. AIRS takes inspiration from a number of elements of the immune system but does not seek to duplicate its functionality precisely. Within [77] AIRS is benchmarked against several classical algorithms including C4.5 which is discussed above and it performs similarly. Within the paper an initial classifier AIRS1 is proposed followed by some improvements to form the AIRS2 classifier. In addition, to reduce the runtime of the algorithm further work was undertaken to parallelise parts of the training process in [114]. Runtime performance is a key issue in industrial contexts where rapid retraining may be required.

CSCA and CLONALG are two immune inspired algorithms that utilise the principles of clonal selection to build classification systems. CLONALG [115] is introduced by Castro and Zuben as an approach that can produce classifiers or with modification be used as an optimisation technique. CSCA is a different algorithm proposed by Brownlee [116] which takes the desirable features of CLONALG and its variants and produces a new classifier on that basis. [116] reports encouraging results for CSCA but precise details are not given.

The Immunos family of classifiers was first proposed by Carter [117] and is unusual in that it was proposed by a medical doctor rather than by a computer scientist or electrical engineer as the rest of the algorithms discussed here are. The original publication does not contain enough information to produce an implementation of the algorithm. In [78]

Brownlee proposes a number of implementations of Immunos-81 (named as Immunos-1 and Immunos-2) which he shows provide similar accuracy to those reported by Carter in the original paper. Brownlee then proposes a new algorithm, Immunos-99, which is a new algorithm inspired by his work with Immunos-81. Immunos-99 reports favourable classification performance on standard data-sets from the UCI machine learning repository. In the course of the EngD work was undertaken to improve the runtime of the Immunos 99 algorithm resulting in an order of magnitude improvement [3] without significantly changing classifier performance.

Since the testing of immune-inspired classification algorithms in this thesis there have been further developments in this field. Tay *et al.* [118] have proposed extensions to the AIRS2 [77] algorithm which includes additional immune-inspired techniques and results in improved classification performance on a number of test data sets.

Another interesting development is the AC-CS algorithm developed by Elsayed *et al.* [119]. This algorithm merges the principles of association rule classification as exhibited by JRip of the classical classification techniques discussed above and immune-inspired techniques to produce a hybrid of the two. The reported performance of the AC-CS classifier is competitive and the algorithm has a large runtime advantage over classical algorithms.

Investigating the impact of these improvements on the business process data set would be an opportunity for further work in this space. The system proposed in Chapter 5 is classifier agnostic so could easily incorporate improved classification algorithms as they become available.

2.4 Summary

This literature review has covered the relevant literature pertaining to business processes, process mining and the principles of autonomic computing. Also surveyed have been existing approaches to incorporating each autonomic principle into a computing and/or process system. None of the literature reviewed has dealt with the generic business process executions. Literature that covered processes was surveyed however these were manufacturing or physical processes covering the production of a physical item and not business processes which are typically more abstract and intangible.

The most promising area of study seemed to be the self-healing of individual process executions. A structural reconfiguration approach may be suitable but only when working at the level of individual business process executions. The detect-diagnose-act cycle used for self-healing systems can then be used to heal individual executions by reorganising their remaining tasks.

The following chapter starts the work by investigating how classical classification algorithms can be used to predict potential process failures as the detect phase of the detect-diagnose-act cycle.

Chapter 3

Classical Classification Techniques for Process Prediction

Initial work on this project focused on the predictive element of a self-correcting system. The specific area being targeted is that of the detect phase of the detect-diagnose-heal cycle used in self-healing systems (see Section 2.2.3). In addition to detection being the first stage, it is also an area that could provide a shorter-term benefit to an enterprise, via the augmentation of existing workflow systems.

Section 3.1 briefly reviews the nature of business process data and describes the approach used to create training data for rest of the thesis. Section 3.2 discusses the setup for classification experiments performed on a sample of real business process data from a large enterprise. Section 3.3 in this chapter discusses the experimental results obtained from classical classification approaches when they are applied to the business process data prepared as discussed.

3.1 Business Process Data and Data Acquisition

Most of the business process data that is available to the project is structured as, or can be transformed to, a data set with a each row representing a single execution of a particular task. The columns present in the data are unique to each business process however data that is suitable for process mining *must* provide the following:

- A Process Instance Identifier which will be used to group the tasks in each process instance together
- Some temporal information to place the task in time. This could be a start time

and an end time, a start time and a duration etc. In some cases only a start time is available, which is acceptable **only** in the case where the business process is known to be sequential.

- A task name (or other identifier) which could be any type providing it can identify the task.

Additional attributes can also be provided for either the process instance (e.g. customer name), or for the specific task instance (e.g. employee number of the executor). Attributes can be of any type. The collection of attributes provided is unique for each business process therefore any analysis methods used must be able to deal with this situation.

Some observed attribute types are:

- Numerical - both integers and real numbers
- Temporal - Timestamp or Date values (which can have varying accuracy depending on the workflow system involved)
- Categorical - Complete lists of possible values are not always available
- Strings - unstructured text, such as notes about the work

One important consideration with regards to the available historical information is that the progression of attribute values is not typically available in the recorded information. If an attribute was to have a number of values throughout the lifetime of the associated task or process instance, only the last assigned value would be available to post-hoc analysis. This lack of intermediate state in the training data may impact the accuracy of predictive methods.

3.1.1 Data Issues

The historical process data stored in a business is not usually without issues. Some of these issues are due to problems in the underlying workflow systems which are not easily solved. For example where updates to task comments after the task is finished cause the start date but not the end date for the task to be updated. Such an error causes the impacted task to appear to end before it starts. For more information on process mining with this data the reader is directed to [1].

The primary issue with any data set is its accuracy. In general process execution data is collected automatically and as such is largely free of keying or transposition errors. It is the case though that workflow systems do not always ensure the consistency of data. One of the most common issues with data is erroneous or missing time stamps; tasks or process data may indicate that a task started but did not finish, or a task finished without being started, or that a task finished before it started. Previous work on process mining (see Section 2.1.5 and [1]) has taken a simple approach where process instances that are tainted in this way are simply discarded. It is not clear how a predictive algorithm would need to deal with such erroneous data.

A closely related issue to that of erroneous times, is that of tasks with zero duration. Zero-duration tasks are usually those which are opened and/or closed in an automated fashion. Zero-duration tasks could also be recorded if the resolution of the timing information is not sufficient to capture the different in start/end times, for example a data set where only the month and year is recorded for start and end times.

Process analysis for business improvement is usually performed after the process instances have been completed. This means that all attributes are available to the analyser. In a predictive capacity however it becomes important to know which attributes are available to the process during execution and which are only assigned at or after the completion of the process instance. For example, in a business process which is resolving faults, the reported type of the fault is available at execution time, but the actual type of fault that was resolved is not available until the end of the process instance.

Some business processes may choose to perform parts of their work in parallel with each other. In one such business process the customer can order redundant services to be provided to ensure service continuity in the event of a fault occurring. These separate circuits would be provided using the same (or very similar) business processes involving the same tasks. Where inferences are to be made between tasks it is important that such separation of tasks is accounted for.

3.1.2 Augmentation by Aperture

Work with business processes in BT is concerned with process analytics and in particular with Process Mining. Process Mining covers methods of reconstructing the executed process design from information recorded during execution. BT's process mining tool, Aperture, has been discussed in [1] and Section 2.1.5.

The use of process mining tools allows the basic data returned from workflow systems to be augmented with causal links between the tasks. The additional information can then be used in predictive models of the system to determine the future actions of the process, or the eventual outcome based on the sequence of completed work.

3.2 Experimental Setup and Measures

3.2.1 Data Acquisition

For the initial experimentation a simple but representative data set was required. The chosen process data is from a high volume, sequentially executing provisioning process from within the BT Group. Task and process attributes were anonymised for confidentiality reasons. The execution traces from this business process were extracted from the workflow systems that manage the execution.

Once the execution traces were acquired they are first processed by the Aperture Process Mining Tool (Section 2.1.5) to reconstruct an appropriate structure for each process instance. The data is then processed again using a script to select categorical attributes from the process and to introduce additional pseudo-attributes regarding timing and structure. The data is then exported in a file format suitable for the tools used. The investigation utilises the Weka data mining toolkit [120]. Weka was chosen because it allows a single input data set to be shared across a large number of classifiers easily. Using identical data sets across classification runs reduces the scope for errors to be introduced into the analysis by data transformations. An additional advantage of Weka was the availability of the implementations of the immune classification algorithms discussed in Chapter 4.

Data exports were produced for the most recent 1, 2 and 3 task sequences for each transition in the test database of process instances. In each case one row appears in the test set for each transition in each process instance. The stages of the process that are shorter than the length for the test set are omitted from the prepared file for that set.

The attributes that comprise the training data are split into two categories. The first set are Process Attributes which comprise information regarding the entire business process. Examples of process attributes would be Customer Name, Product, Geographic Location and alike. Task Attributes are those which are specific to a task within the business process. Examples of task attributes are the employee performing the task, the outcome of a line test and alike.

Some of the task and process attributes might be timestamps. Rather than include the raw timestamp as an attribute, which is unlikely to be directly useful, each timestamp is instead decomposed into the following elements.

- The day the timestamp occurs on.
- The AM or PM specifier.

- The percentile of the time within the execution day.

These elements are used to attempt to allow the classifier to discover relationships between the nature of the timestamps and the outcome as part of the training process. As a result of the decomposition the classification algorithms do not need to be aware of timestamps as a data type.

Further attributes are added to encode the temporal position of the 1, 2, or 3 step sequence of tasks within the business process. The attributes used for this are:

- Time in seconds from the start of the business process to the start of this sequence.
- For each task:
  - Time in seconds from the start of the process to the task start.
  - Time in seconds from the start of the sequence to the task start.
  - Task duration in seconds.

Finally, for each task the number of tasks connecting into this task in the process execution graph is included as the in-degree. In sequential processes this will always be 1, since each process will have one and one only immediately previous task. In parallel processes this can be greater than one, for example in the scenario where task C starts once the concurrently running tasks A and B both complete.

For the initial experimentation the data set produced was further filtered to include process executions containing only the most frequent 10 tasks. This simplified data set is known throughout as the 10-task data set. This sample data has 8 input process attribute columns and 11 columns for each task. The exact structure of the sample data is shown in Table 3.1. Using a simplified data set will allow for faster initial testing, and mitigate the effect of the extreme amount of noise present in the source data. The distribution of success (outcome=Y) and failure (outcome=N) for each of the three test sets is shown in Figure 3.1.

In addition to the noise typically present in a business process data set, there is also an element of class skew, although the degree of this skew varies between different business processes. Figure 3.1 clearly shows the unbalanced nature of the classes in the test data set used in this section. This class skew did influence the thought process used to choose the classifiers to test in this section, however given the industrial nature of the EngD there was also a strong requirement that any techniques chosen were acceptable and appropriate for the industrial partner. Therefore, Decision Trees and Association Rules where chosen for initial testing because they have a track record of usage at BT for

Table 3.1: Structure of the classification training data for the simplified data set with a single recent task. Columns 9 to 20 would be replicated for the each of the 1,2 or 3 tasks in task sequence included in the data.

	Column Name	Type/Description
1	CARE LEVEL	Categorical - Process Specific
2	COMMS PROVIDER	Categorical - Process Specific
3	GM	Categorical - Process Specific
4	PRODUCT	Categorical - Process Specific
5	REGION	Categorical - Process Specific
6	SOM	Categorical - Process Specific
7	SUB PRODUCT	Categorical - Process Specific
8	SUB ZONE	Categorical - Process Specific
9	Task 1 Def ID	Categorical - identifying the type of task performed
10	Task 1 Start Day	Categorical - The day of the week Task 1 began on
11	Task 1 End Day	Categorical - The day of the week Task 1 ended on
12	Task 1 Start AM PM	Categorical - Whether Task 1 started in AM or PM
13	Task 1 End AM PM	Categorical - Whether Task 1 ended in AM or PM
14	Task 1 Start Percentile	Numeric - Which percentile of the day did the task start in
15	Task 1 End Percentile	Numeric - Which percentile of the day did the task end in
16	Task 1 In Degree	Numeric - How many previous tasks link to this on
17	Task 1 Out Degree	Numeric - How many following tasks link from this one
18	Seconds to task 1 start	Numeric - How many seconds elapsed between the start of the business process and the start of task 1
19	Seconds frm sequence start to task 1	Numeric - How many seconds elapsed between the start of this records task sequence and the start of task 1
20	Task 1 Duration Seconds	Numeric - How long task 1 ran for (task end - task start)
21	Depth	Numeric - How many previous tasks exist for this process execution
22	OUTCOME	Categorical - Was this business process completed successfully (Y or N)

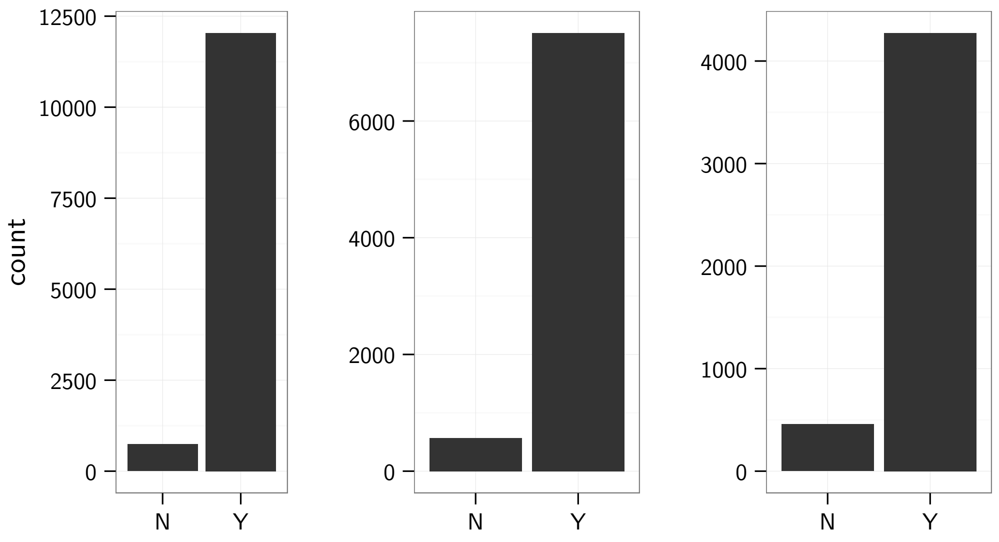

Figure 3.1: Histograms showing distribution of outcomes in the 3 sample sets

tasks involving skewed data such as customer churn prediction and network failure event detection. The results of this testing were then used to justify the testing of other less familiar classification techniques in Chapter 4.

3.2.2 Measurement

In order to compare the performance of the algorithms under test three measures common to machine learning will be used. All tests will use a 10 fold cross-validation approach on the data set. The first measure is the overall accuracy of the algorithm which is the number of correctly predicted values as a percentage of the records in the data set. The second measure is the true positive (TP) rate for the outcome classes which gives the proportion of the testing instances that correctly lie in that class (for each class). The third measure is the false positive (FP) rate which gives the proportion of the predicted results which incorrectly appear in this class (for each class).

As discussed in Section 1.3.1, headline accuracy is not the best measure for deciding the business suitability of the algorithms. The True-Positive and False-Positive rates are more important from a cost-benefit perspective. Analysis and conclusions will be performed with this in mind.

3.3 Experimental Results

The algorithms tested were Decision Trees and Association Rules. These were chosen because they are well known algorithms suited to categorical classification problems. Further, they are both have a excellent track record of successful uses within the industrial partner. Results for each technique are presented in the remainder of this section.

3.3.1 Decision Trees

The J48 classifier was used for the decision tree experiments. J48 is a java based implementation of the C4.5 decision tree algorithm [106] provided with the Weka toolkit [120]. J48 was chosen for business suitability reasons. J48 and decision trees in general are well known and well used inside the industrial partner, using a technique which is already understood means there is would likely be less resistance to implementation of a solution involving these techniques rather than something less well known. Unlike Naive Bayes J48 does not make any assumptions regarding the interdependence of input variables, this is an advantage because when dealing with process data it is not always clear which attributes are in fact dependent on each other. A further advantage of decision trees is that they produce human-understandable trees and/or rule sets which can provide information for manual process improvement activities.

Since J48 is an implementation of the C4.5 algorithm it performs classification identically to C4.5. First a tree is grown by recursively spitting the data set by whichever variable best split the set of training samples into subsets by class. The measure used in J48 to determine the split is normalised information gain which means the attribute with the highest information gain is chosen to split the training set at each stage.

Classification is performed by navigating the decision tree by starting at the root and comparing the attribute value from the instance to the values associated with split point on the node, and continuing to the next node. This continues until a leaf node is reached when the label of the leaf is returned as the class for the new instance.

There are a number of parameters used to control the J48 classifier. Those used for the experiments in this section are shown in Table 3.2. The results of applying the J48 decision tree algorithm to the prepared data are shown in Tables 3.4, 3.5 and 3.6.

The imbalance between the class frequencies in the test data set appeared to be causing the decision tree algorithm to simply ignore the failures and return a default prediction of success. This is not acceptable in the target environment so techniques to address this issue were investigated. An approach to aid classifiers in dealing with unbalanced data sets is the MetaCost approach [111] which can extend an existing classifier with cost sensitivity. The MetaCost approach was the one investigated in this work via the `CostSensitiveClassifier` class in the Weka toolkit [120].

The MetaCost [111] approach treats the underlying classifier as a black-box requiring no modifications to the underlying algorithm. The alternative approach is to use stratification which adjusts the training data to adjust the weighting of each class by either under or oversampling but this either reduces the data available to learning or increases the learning time in the case of oversampling. Instead, MetaCost creates a number of re-samples of the training data, uses the classifier to build an ensemble of models trained on those re-samples, then relabels each instance in the original training data with the highest probability class resulting from applying the ensemble set of classifiers. Full details are presented in Domingos's paper [111]. MetaCost allows the J48 classifier to take into account the higher cost of not predicting a process failure, which is an essential feature of the business process being tested. Weka's implementation of MetaCost, `CostSensitiveClassifier`, has a number of parameters which adjust its operation, those used for the experiments in this section are give in Table 3.3.

Using the J48 classifier augmented with MetaCost here is a clear and expected increase in both true and false positive rates for increasing cost of misclassification. It is also evident that increasing costs above 10 in each of the three test sets has an unacceptable effect on false positive rate. The tests with cost of misclassification of 10 have excellent results. Another interesting result is the decrease in true positive rate when additional process steps are incorporated into the training set, this suggests that the process structure is less important to success than the categorical attributes of the process.

Incorporating a penalty for misclassifying the more important failure cases has a clear improvement upon the resulting true positive rate for that class of faults with a minimal impact on the false positive rate of the larger success class.

The results for the unmodified decision trees and the MetaCost augmented classifiers at cost 5 are very similar and would likely be acceptable for industrial use due to their very low false positive rates, however the true positive rates of between 44% and 59% would mean that a large number of failures would potentially go undetected. The best performing classifier from all of the runs is that with a cost of a missed failure at 10 and 1-step of process history which shows excellent performance with 86.5% of failures being detected with only a 2.8% false positive rate. The remaining classifiers have unacceptable performance for the industrial context. It is interesting that the best performing classifier only contains the details for the most recent task in the process instance which suggests that the increasing number of tasks combinations observed as the number of tasks included in the prediction data increases is making it difficult for the training process to produce a highly performing model.

Table 3.2: Parameters used for J48 classifier runs

Parameter	Value
Binary Splits	false
Confidence Factor	0.25
Debug	false
Min Observations per Leaf	2
Data Folds	3
Reduced Error Pruning	false
Save data for Visualisation	false
Sub Tree Raising (pruning)	true
Unpruned	false
Smooth leaf counts with Laplace	false

Table 3.3: Parameters used for CostSensitiveClassifier augmented classifier runs

Parameter	Value
Classifier	weka.classifiers.trees.J48 (with parameters from Table 3.2)
Cost Matrix	[0.0 $COST; 1.0 0.0] (with $COST set per the run)
Debug	false
Minimise Expected Cost	true

Table 3.4: Results of applying J48 to the 1-step test set

Classifier	Accuracy	True Positive Rate	False Positive Rate
J48 [106]	96.580%	44.6%	0.2%
J48 (with cost 5)	96.494%	44.5%	0.4%
J48 (with cost 10)	96.533%	86.5%	2.8%
J48 (with cost 100)	88.786%	94.9%	11.6%

Table 3.5: Results of applying J48 to the 2-step test set

Classifier	Accuracy	True Positive Rate	False Positive Rate
J48 [106]	95.901%	47.3%	0.4%
J48 (with cost 5)	95.901%	50.4%	0.7%
J48 (with cost 10)	84.309%	78.6%	15.3%
J48 (with cost 100)	10.736%	97.7%	95.8%

Table 3.6: Results of applying J48 to the 3-step test set

Classifier	Accuracy	True Positive Rate	False Positive Rate
J48 [106]	95.076%	57.2%	0.9%
J48 (with cost 5)	94.907%	59.0%	1.2%
J48 (with cost 10)	83.368%	77.7%	16.0%
J48 (with cost 100)	15.025%	97.2%	93.8%

3.3.2 Association Rules

Another machine learning technique that is applicable to predictive problems is association rule mining e.g. JRip [109]. A rule mining classifier attempts to apply association rules mined from a training set to the classification of the test set of data. Business process data is generally both noisy and somewhat inconsistent. The exact results of applying the JRip association rules algorithm are shown in Table 3.8. The parameters used for these runs of JRip are specified in Table 3.7.

In terms of the true positive rate for failure detection the rule based classifier is not significantly different than the decision tree cases with rates between 32% - 58% overlapping with the TP rates for the decision tree classifiers. The false positive rates though are higher than those of the decision trees which makes the rule classifier less palatable to the industrial partner. In the industrial context false positive rate would be the key driver of additional cost to the business, therefore techniques that deliver lower false positive rates are to be preferred.

An interesting characteristic of the results of the rule classifier is that they gain accuracy with additional process data which is counter to the decision tree approach. This appears to be due to the ability to infer more specific rules to deal with individual cases in the training set. In a more realistic data set highly specific rules are likely to have lower predictive power due to over fitting. Further testing on disjoint training and testing sets, and testing on other process data could be used to gain further insight into this effect. Internal testing on other process data at BT [4] suggests that in less skewed data sets the predictive power of the rules is significantly diminished.

Table 3.7: Parameters used for association rule classifier runs

Parameter	Value
Check Error Rate as stop-criterion	true
Debug	false
Data Folds	3
Min Total Weight of Instances	2.0
Optimisation Runs	2
Pruning	true

Table 3.8: JRip Classification performance on the test data sets

Data Set	Accuracy	True Positive Rate	False Positive Rate
1-Step	94.828%	32.8%	1.4%
2-Step	94.328%	42.9%	1.8%
3-Step	92.942%	58.1%	3.3%

3.4 Conclusion

This chapter examines the performance of two classical classification algorithms, Decision Trees and Association Rules, in terms of their ability to predict a business process outcome. Association rules are outperformed by decision trees and so are discounted from further testing. The results obtained show that decision trees can be used to produce an accurate predictive classifier, however there is a need to tune the parameters of MetaCost to counter the skewed nature of the input data. This tuning might lead to manual effort however it would also be possible to use optimisation techniques such as genetic programming [121] to determine appropriate parameters for MetaCost. Such an optimisation would attempt to minimise the cost of the classification results based on the impact of each type of failure to the business; true positives have a small cost impact consisting only of the resolution cost with no cost due to customer engagement, false positives have a high cost as they incur both the cost of rectifying the problem and the reputation damage of failing to meet the customers expectations. Therefore a suitable cost function might be $\text{Cost} = (\text{£ per TP} \times \text{Number of TP}) + (\text{£ per FP} \times \text{ Number of FP})$. Exact values for the monetary impacts would be obtained from the business.

To attempt to lessen or remove the requirement for parameter tuning testing alternative classification approaches was desirable. The alternatives tested were those based on artificial immune systems. These techniques were attractive because of the cellular nature of their construction, it seemed likely they would be well placed to detect and classify the highly variable failure cases in the business process data. Experiments to determine the performance of the artificial immune systems algorithms in the business process outcome scenario are discussed in the following chapter.

Chapter 4

Immune Inspired Techniques for Process Prediction

4.1 Introduction

In the previous chapter a number of classical classifiers were tested. This testing revealed weaknesses for the unmodified algorithms when they encountered skewed input data, such as typical business process data. Therefore some investigation of alternative techniques was desirable. In this chapter a number of alternative classifiers inspired by artificial immune systems are tested against business process data. The experiments and results are discussed and then evaluated against the benchmark set by the classical classification algorithms. Additionally Section 4.6 discusses the essentials of an internal case study where the Immunos 99 classifier is tested against an additional data set.

When business process executions fail, they usually do so in ways that are not well understood. This means that it is not easy to partition the problem space into clear success and failure parts. Decision trees attempt to split the problem space by choosing which attribute best partitions the input into classes, this is seems to be at odds with the aforementioned nature of business process systems. Artificial Immune Systems inspired classifiers however are based upon immune systems in nature which evolve sets of individual recognition cells each of which captures information about the learned environment independently. Given that each of these cells can evolve to match very different scenarios, this seems a good match for the business process space where failures can occur for an incredible variety of different reasons.

Each of the specific algorithms used is introduced briefly in the following section.

4.2 Candidate Algorithms

4.2.1 AIRS

AIRS [77] produces a classifier by developing a pool of memory cells which have the function of matching specific observations (antigens) as they do in a mammalian immune system. At a very high-level, AIRS operates as per Algorithm 2.

The initialisation step (line 1) normalises the values in the training data such that the distance between any two attribute values is between $[0, 1]$. This step also ensures that the range of possible values for cell-to-cell interactions are in the range $[0, 1]$. Next, the affinity threshold is calculated as the average affinity value over all the antigens feature vectors. Affinity is the measure of similarity between 2 feature vectors with small values representing strong affinity. The calculated affinity threshold is used later in the algorithm. Finally, the initial population of memory-cells and Artificial Recognition Balls are seeded using random sampling from the training set. An Artificial Recognition Ball (ARB) is a construct that groups an antibody, the number of resources held by the ARB and the current stimulation value for the ARB.

Memory cell identification and ARB generation (line 3) starts by selecting the existing memory cell (mc_{match}) that is most stimulated by the current antigen ag, where stimulation is calculated as $1 - \text{affinity}(ag, mc_{match})$. Once the memory cell is identified new ARBs are created and placed into the population of (possibly existing) ARBs. Precise details on the exact generation of ARBs are given at [77, p. 298].

Competition for resource and development of candidate memory cells (line 4) is a phase where the ARBs that are currently present in the pool are evaluated to determine their fitness and if they are to be retained into the next iteration of the algorithm. Fitness of a given ARB is calculated by first determining the stimulation level for all ARBs given the current antigen, ag. The stimulation value is then normalised across the ARBs in the population that have the same class as ag. Each ARB is then allocated a slice of the available resources. If the level of resource allocation exceeds that indicated by the

Algorithm 2: High-Level pseudo-code description of the AIRS Algorithm

```
1 Initialisation ;
2 foreach antigen (ag) do
3     Memory cell identification and ARB generation ;
4     Competition for resources and development of candidate memory cell ;
5     Memory cell introduction ;
6 end
7 return memory pool for classification
```

resource limit parameter then resources are removed from the weakest ARBs until the resource limit is satisfied. Any ARBs with zero resources are then deleted. To determine if the ARBs have been stimulated enough by the current antigen the algorithm uses a stopping criterion. If sufficient stimulation has occurred (based on parameter value) then the algorithm moves onto the next antigen, otherwise the ARB is given the opportunity to produce mutated offspring and this step is repeated.

Memory cell introduction (line 5) is the final stage in the training algorithm for the current antigen. This stage is where the decision is taken on whether to introduce the memory cell being developed here into the memory pool. The memory cell identified in line 3, mc_{match} is replaced with the new memory cell $mc_{candidate}$ where the new cell is stimulated more by the antigen and the affinity between the 2 memory cells is below the threshold specified by the affinity threshold calculated on line 1 and the affinity threshold scalar parameter. This completes the processing for the current antigen.

Once all antigens have been processed, the resulting memory cells can be used for classification. AIRS classifies data by using a k-nearest-neighbour approach where the classification is determined by the majority vote of the closest k memory cells to the observation.

Key parameters for the AIRS algorithm are:

- Distance metric - This is used the calculate the distance and affinity between observations and between observations and memory cells. Euclidean distance is typically used.
- Resource limit for pools of candidate memory cells. AIRS assigns resources to potential memory cells as part of its candidate selection process. The weakest cells are removed until the total candidate population is below this threshold.
- Affinity Threshold Scalar - This determines if a new entrant to the memory pool should remove the memory cell that was previously the best match for the current observation or not. Removal helps to keep the memory pool diverse by avoiding the retention of similar memory cells.

The exact parameters used during the testing of AIRS in this section are give in Table 4.1.

Table 4.1: Parameters used for testing of the AIRS algorithm in Chapter 4

Parameter	Value
Affinity Threshold Scalar (-F)	0.2
Clonal Rate (-C)	10.0
Hypermutation Rate (-H)	2.0
Mutation Rate for Cloned ARBs (-M)	0.1
Total allocatable resources (-R)	150
Stimulation threshold (-V)	0.9
Number of antigens used to calculate the affinity threshold (-A)	-1 (all)
Initial ARB pool size (-B)	1
Initial memory cell pool size (-E)	1
Number of nearest neighbours for classification (-K)	3
Random Seed (-S)	1 though 10

4.2.2 CLONALG

CLONALG has high level similarities with AIRS in that CLONALG also involves the development of a pool of memory cells which will form the final classifiers. However there are very different algorithms. CLONALG does not include mechanisms of affinity mutation and somatic hypermutation [114] so will evolve differently to the AIRS algorithm. CLONALG also runs for multiple generations of memory-cells whereas AIRS uses a single generation. A high level description of the algorithm in pseudo-code is shown in Algorithm 3. The steps are briefly discussed below.

The initial stages of the CLONALG algorithm are straightforward. First (line 1) the antibody pools that will form the final classifier are created. Unlike AIRS, there is no pre-processing or normalisation step. The rest of the training algorithm is then a nested loop over each generation and example from the training set (antigen) in turn (lines 2 and 3). Line 4 shows that the training set is processed in a random order via the selection without replacement step. Lines 5 and 6 are straightforward.

The Clone AB step on line 7 makes copies of the antibodies in AB in proportion to their affinities from AF. This means that the antibodies that show the highest affinity to the current antigen become more frequent and are therefore more likely to survive through the remaining steps of the algorithm.

In the affinity maturation step (line 8) the input antibodies (AB_{cloned}) to the step are mutated, with the level of mutation being proportional to the affinity from AF, the greater the affinity between the antibody and the antigen the less mutation is applied to the cloned antibodies in AB_{cloned}. This results in the antibodies in AB_{cloned} being updated with the newly mutated versions.

Algorithm 3: High-Level pseudo-code description of the CLONALG Algorithm

```
1  Initialisation ;
2  foreach Generation (g) in $[0, G]$ do
3      while Training set is not empty do
4          $ag$ := randomly (without replacement) remove antigen ($ag$) from training
            set ;
5          $AF$ := Calculate affinities between each $ab \in AB$ and $ag$ ;
6          $AB$ := select $n$ antibodies from the system with highest affinity with $ag$ ;
7          $AB_{cloned}$ := Clone($AB$);
8          Affinity Maturation($AB_{cloned}$) ;
9          $AF_{cloned}$ := Calculate affinities between each $ab \in AB_{cloned}$ and $ag$ ;
10         Insert new memory cells into pool ;
11         Insert random antibodies into pool to the replace $d$ antibodies with the
            lowest affinity ;
12     end
13 end
14 return Antibodies from memory pool ;
```

At line 9, the cloned antibodies are exposed to the antigen in the exactly the same way as the original antibody set was on line 5. Once the affinities are calculated, then new antibodies from AB_{cloned} can be inserted into the memory pool. Antibodies in AB_{cloned} are placed into the memory pool where the affinity of the new antibody to the current antigen is higher than the affinity of the most closely matching antibody in the memory pool, the old antibody is replaced in the pool by the new one. In addition on line 11, the replacement step inserts new random antibodies into the memory pool in place of the d antibodies with the lowest affinity with ag to ensure a level of diversity in the memory pool.

The final classifier returned on line 14 consists of the memory pool at the conclusion of the training algorithm. CLONALG is then able to classify new observations by returning the classification of the best matching memory cell from the memory pool.

The exact parameters used during the testing of CLONALG in this section are give in Table 4.2.

Table 4.2: Parameters used for testing of the CLONALG algorithm in Chapter 4

Parameter	Value
Antibody Pool Size (-N)	30
Clonal Factor (-B)	0.1
Selection Pool Size (-n)	20
Total Replacements (-D)	0 (disabled)
Number of Generations (-G)	10
Random Seed (-S)	1 though 10

4.2.3 CSCA

CSCA is introduced in [116] by Brownlee as a technique inspired by Clonal Selection and CLONALG that represents the classification problem as one of function optimisation. In CSCA the memory pool population is exposed to partitions of the input data a single partition at a time. Each memory cell keeps an internal score representing its affinity to the training instances and mutation is applied in inverse proportion to the affinity of the memory cell. One feature of CSCA that is not present in AIRS or CLONALG is its ability to automatically set the required number of memory cells.

A high level description of the algorithm in pseudo-code is shown in Algorithm 4. The steps are briefly discussed below.

The first step in the algorithm (line 1) is to populate the antibody pool using randomly selected antigens from the training set (without replacement). Once initialisation is completed the next steps of the algorithm are repeated for the required number of generations (G). Within each generation a number of steps are required (lines 3-6).

First, the affinity scores for each antibody and antigen combination is calculated and the fitness of each antibody is then calculated. In the selection step (line 4), the antibodies to be carried forward to the following steps are selected. The selected set includes all antibodies unless: they have a misclassification score of 0 or the antibodies have a fitness score below ϵ (these low scorers are also removed completely from the base antibody population).

In the cloning and mutation stage (line 5) the antibodies that were selected on line 4 are subjected to a proportionate cloning and mutation stage to generate a modified antibody set. Full details of this set are available in [116]. Finally for this generation the generated clones are inserted into the main antibody pool. In addition, n randomly selected antigens are also inserted where $n = |AB_{selected}|$.

Algorithm 4: High-Level pseudo-code description of the CSCA Algorithm

```
1 Initialise antibody pool ;
2 foreach Generation (g) in $[0, G]$ do
3     Calculate affinities between all antibodies and antigens ;
4     $AB_{selected}$ := Selection ;
5     Cloning and mutation ;
6     Clones and new antibody insertion ;
7 end
8 Final pruning ;
9 return Antibodies from memory pool ;
```

Table 4.3: Parameters used for testing of the CSCA algorithm in Chapter 4

Parameter	Value
Initial Population Size (-S)	50
Generations (-G)	5
Clonal Scale Factor (-a)	1.0
Minimum fitness threshold, ϵ (-E)	1.0
Number of neighbours for classification (-k)	1
Total partitions (-p)	1
Random Seed (-r)	1 though 10

Unlike AIRS and CLONALG the antibody set is not immediately returned for use as a classifier, instead CSCA includes a final pruning step (line 8). In the final pruning step the antibody population is again exposed to all antigens from the training data and the affinity and fitness scores calculated for the antibodies within. A selection step identical to that on line 4 is performed to select a set of antibodies. These selected antibodies are then returned for use in classification. CSCA then makes classifications using the same k-nearest-neighbour approach as AIRS and its set of selected antibodies.

Key parameters for CSCA would be:

- The number of generations in the training phase.
- The initial memory pool population size.
- The minimum fitness threshold - which controls the removal of poorly performing memory cells from the pool.

4.2.4 Immunos 99

Immunos 99 [78] is a newer algorithm based on the principles of Immunos-81 [117]. Immunos 99 differs from the other techniques in that it uses competition between different memory cells at classification time rather that at training time. Immunos 99 also utilises a single exposure to training data as per AIRS which is desirable from a runtime perspective. Immunos 99 works by developing a memory pool of memory cells for each classification label which contains cells which correlate strongly with one classification label and weakly with the others. A high level description of the algorithm in pseudo-code is shown in Algorithm 5. The steps are briefly discussed below.

Immunos 99 starts by partitioning the training data (antigens) into groups based on the classification label attached to each antigen. The loop starting at line 2 then produces

Algorithm 5: High-Level pseudo-code description of the Immunos 99 Algorithm

```
1  Divide the training data into antigen groups (per class) ;
2  foreach Group (g) do
3      Create initial b-cell population ;
4      foreach Generation (G) do
5          Calculate affinities between antigens in $g$ and b-cells in $g$ ;
6          Perform population pruning ;
7          Affinity Maturation ;
8          Insert random antigens from $g$ ;
9      end
10 end
11 foreach Group (g) do
12     Final population pruning ;
13 end
14 return the final b-cell population ;
```

a set of b-cells for each group independently. The first set of b-cells are copies of the antigens of the group.

The training for each group starts by calculating the affinities between the all antigens in all groups and the current set of b-cells. For each antigen the b-cells are ranked by their affinities to that antigen and the rank of each b-cell is recorded. A fitness measure for each b-cell is produced by dividing the sum of ranks for the antigens in the same group divided by the sum of ranks for the antigens in the other groups. This gives each b-cell a fitness score representing how well it differentiates the current group from the other groups. Full details of this are available at [78, p. 18].

Once the affinities and fitness scores are known, then the population is pruned (line 6). Population pruning at this stage involves removing those b-cells that have fitness score between 0 and 1 which means that they are more closely related to antigens in other groups. The exact fitness value used for pruning is determined by a parameter ϵ similar to CSCA.

In the affinity maturation step (line 7) the population of b-cells is ordered by fitness score in descending order. Each cell is then cloned using its rank-proportion of the total number of antigens belonging to the b-cells class from the training set. Once the clones are created they are mutated inversely proportional to the b-cells rank so that larger mutations occur to less fit cells. After these new b-cells are introduced to the population a number of randomly selected antigens from the current group are added. The number of random cells introduced is the same as the number removed by pruning. This introduces additional diversity into the population.

Once all of the groups have been processed the final pruning step can be performed.

Table 4.4: Parameters used for testing of the CSCA algorithm in Chapter 4

Parameter	Value
Generations (-G)	1
Random Seed (-r)	1 though 10
Minimum Fitness Threshold, ϵ (-E)	-1.0
Seed population ratio (-S)	0.2

Each group of b-cells is then exposed again to the entire antigen set. However, instead of accumulating rank based scores as before this time only the best matching cell is scored. Pruning then proceeds exactly as before removing those cells with fitness scores between $[0, \epsilon]$. The final b-cell population can then be returned for classification.

Classification with Immunos 99 is slightly more complicated than for the other algorithms in that there is an element of competition between the memory cells in each classification label group. The new observation is exposed to each population of memory cells and the label for the pool with the highest avidity to the new observation is the classification label applied. Avidity is defined as the sum of the affinities for the memory pool.

An additional point in Immunos 99's favour for the desired parameter free classifier is that Brownlee notes a general insensitivity to changes in its parameters. Given this insensitivity the parameters are not discussed further here. Interested readers are directed to Brownlee's description of the algorithm [78]

4.3 Experimental Setup

The hypothesis for this chapter is that immune inspired algorithms will perform better for business process prediction than the classical algorithms discussed in Chapter 3. If this was the case then they would be the obvious choice for utilisation in an autonomic process system.

In order to test this the immune inspired algorithms were exposed to the same data sets used for the classical classification techniques. Full details on the data preparation are included in Section 3.2. Each of the 4 immune classifiers were exposed to the 1, 2, and 3 step sets of the same process data used in Chapter 3. Each classifier was tested using a 10-fold cross-validation technique and the results collected for comparative analysis, the experiments were repeated 10 times each. Utilising the same test data and techniques for both set of classifiers will allow the relative performances to be compared.

The software used for these experiments was the immune algorithms for Weka [120]

Table 4.5: Table summarising results of applying the immune classifiers. The 10 repetitions of each experiment are summarised using median.

Classifier	Steps	Accuracy (%)	TP (%)	FP (%)	TN (%)	FN (%)
AIRS	1	92.99	25.14	2.84	97.16	74.86
AIRS	2	91.24	20.76	3.37	96.63	79.24
AIRS	3	88.69	22.05	4.06	95.94	77.95
CLONALG	1	90.78	7.16	2.84	97.16	92.84
CLONALG	2	87.78	10.78	3.37	96.63	89.22
CLONALG	3	84.26	15.28	4.06	95.94	84.72
CSCA	1	94.17	0.07	0.05	99.95	99.93
CSCA	2	92.93	0.18	0.07	99.93	99.82
CSCA	3	90.14	2.29	0.48	99.52	97.71
Immunos 99	1	84.65	68.11	14.35	85.65	31.89
Immunos 99	2	84.11	73.94	15.10	84.90	26.06
Immunos 99	3	80.23	70.96	18.81	81.19	29.04

package [116, Section 8][1], an open source implementation of a number of immune algorithms including those discussed.

4.4 Results

Results from the application of these algorithms to the described test data are found in this section. Box-plots showing the distribution of the True Positive, False Positive and Accuracy for each combination or Classifier and Data Set are shown in Figures 4.1, 4.2 and 4.3 respectively. The median performance statistics for each combination of data set and classifier are shown in Table 4.5.

As can be expected from testing a wide variety of techniques, the performance of the immune inspired algorithms on the test data is varied. Of the immune inspired algorithms tested Immunos 99 [78], [117] achieved the best performance in terms of the true positive rate, the false positive rate is however high enough to be a concern (see section 3.2.2).

The CSCA algorithm shows some surprising results with extremely low true and false positive rates reported for all test sets. Results of approximately 0% for both true and false positive rates suggests that CSCA is making a default outcome=Y prediction in virtually all cases. Also notable in this context is the overall performance of CLONALG which is much more variable between runs that the other classifiers suggesting that the randomness inherent in the algorithm has a larger impact that for the other algorithms. CLONALGs performance is also poor with TP rates between 7% and 15%.

[1]Available at: `http://wekaclassalgos.sf.net`

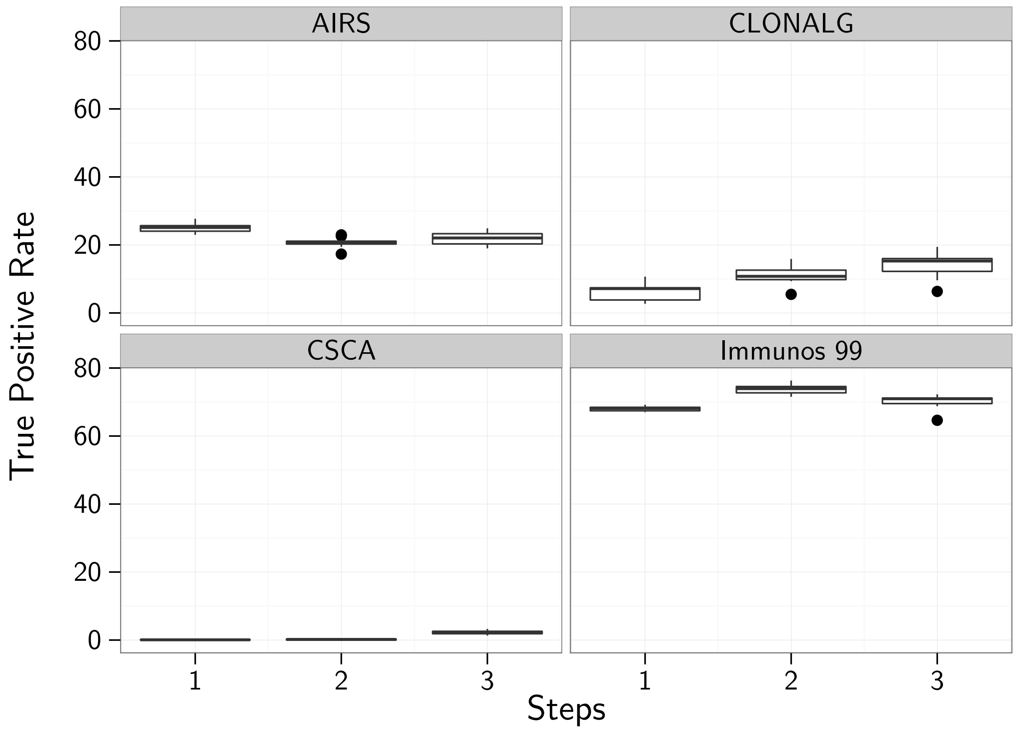

Figure 4.1: Box-plot showing the True Positive Rates for each run of the Immune Classifiers, grouped by classifier and number of steps in the data-set

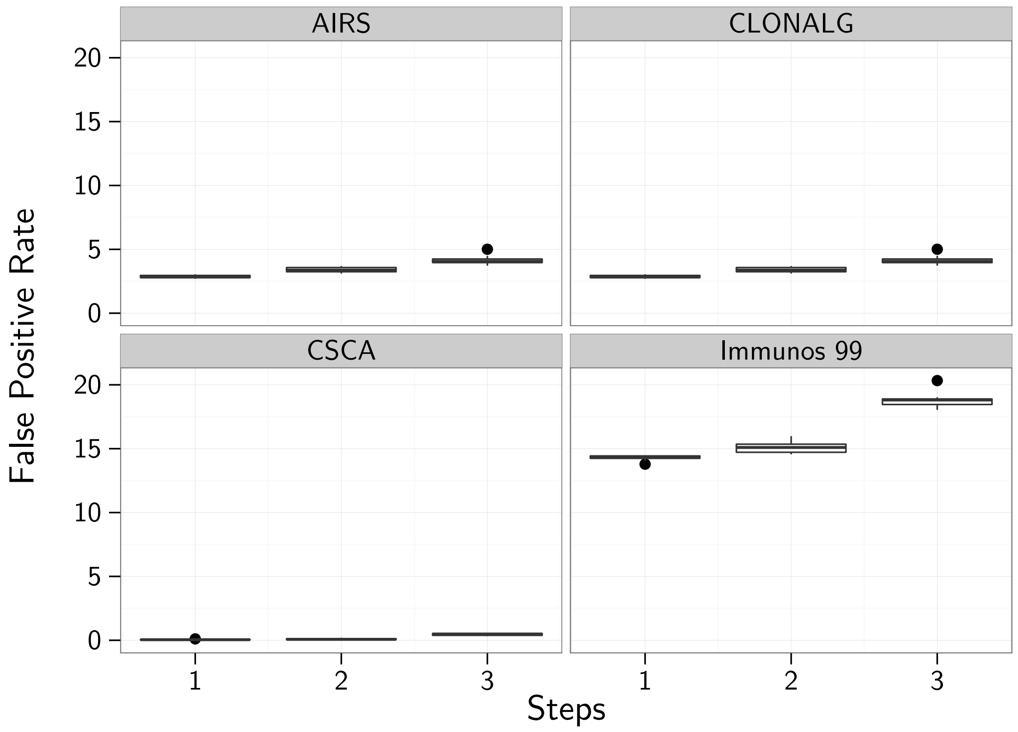

Figure 4.2: Box-plot showing the False Positive Rate for each run of the Immune Classifiers, grouped by classifier and number of steps in the data-set

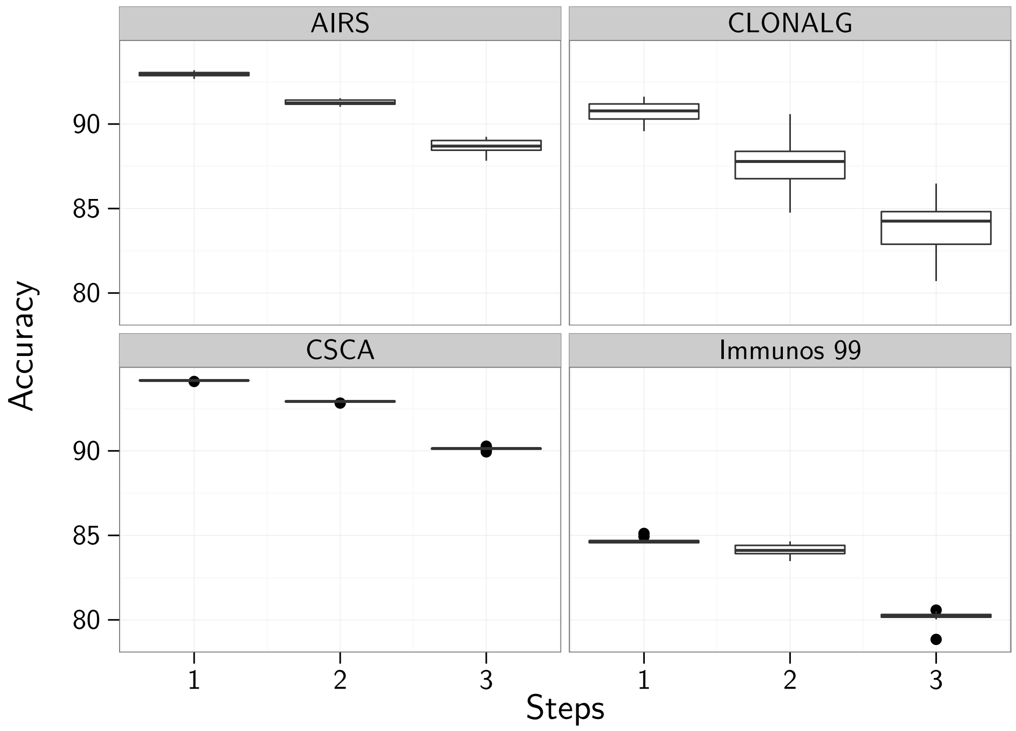

Figure 4.3: Box-plot showing the overall accuracy for each run of the Immune Classifiers, grouped by classifier and number of steps in the data-set

The results for these techniques prompt questions on potential reasons for such a wide variation. Further research is required on the techniques to see if useful insight can be gained from the differences in the techniques. For example, could the variation be related to the different distance metrics used in the different classifiers, or different internal representations.

As with the classical classifiers, further testing on more complex data sets is required to determine if these performance levels can be sustained on more complex input data. An internal BT case study using the Immunos 99 algorithm on a different data set is discussed in Section 4.6.

4.5 Conclusions from Initial Experiments

In this section a number of immune inspired classification approaches have been investigated to determine if they offer advantages over the classical approaches covered in Chapter 3

Of the immune approaches Immunos 99 has the most favourable performance characteristics although its false positive rate is high enough to be a concern. Further work would be required to look further at the specifics of the algorithms that have been tested to discover the reasons for differences in performance. A particularly important area for investigation will be the representation of the business process instances, both in terms of the data provided to the classifiers for training and testing, and in terms of the internal representation and measures used by the classifiers themselves.

Previous investigation internally at BT suggests that the performance of predictive algorithms will degrade rapidly with additional complexity [4]. Investigation is also required to determine if results reported here are transferable to other more complex process data sets.

4.6 Case Study: High-Volume Wholesale Service Provision

A later opportunity brought a new data set into consideration due to interest in the immune-inspired classification results at the industrial partner. This data set comprises of the process execution history for the provision of a very large number telephony services (of the order of $100,000$'s). The product involved is of strategic importance and has a material effect on the performance of the company. The volume of data and criticality of the impacted product provided an ideal opportunity to test the immune inspired predictive approach described in this chapter.

The content of this section is adapted from an internal BT document written by the author originally intended for internal consumption [5]. This section is intended to see if the performance of the Immunos 99 classifier is consistent between data sets, given the initially promising results from the previous section and interest from BT in those results. This case study is not intended to confirm or deny that Immunos 99 is the best classifier for all process data. Additionally, the long run time of the other classifiers meant that was not possible to test them in the time scale given by the industrial partner for the production of the internal report. The Immunos 99 classifier used in this section is the parallelised implementation described in [3] which has a considerable runtime advantage over the other algorithms, and over the original Immunos 99 implementation used in the previous section.

4.6.1 Experimental Setup

The data set contains all of the provision orders for the impacted service arriving within a 1 month period. The exact number of provisions is confidential. The success criteria for each process is whether the order was completed on or before the Customer Commitment

Date (CCD), which is the date agreed with the customer for service commencement.

The data set consists of >700,000 observations of changing process state during the life time of these provision orders. There are approximately 200 different tasks that have been observed in the data sample. The data set was prepared in the same way as the data sets discussed previously in the thesis, see Section 3.2.1 for more details.

4.6.2 Experiments

The first step was to determine how the Immunos 99 classifier performed on the extracted data with no additional modification or mitigating steps being used. Unfortunately it was not possible to train the classifier on the entire data set because training the algorithm resulted in the process exceeding the 16GB of memory available to the test machine. Instead, to get a representative results, a sample of 100,000 observations was taken to be used in the experimentation. The results of a 10-fold cross validation in terms of the performance of the classifier was as follows (raw results from Weka are shown in Appendix A):

- Overall accuracy: 75% - This is a simple measure measurement of the number of observations the classifier classified correctly.
- True Positive Rate (TP): 57% - This is the percentage of process observations from process failures that were correctly identified by the classifier. This is analogous to a correct predication of CCD failure.
- True Negative Rate (TN): 77% - This is the percentage of observations of CCD Successes that are correctly identified by the classifier. These are cases where no intervention was suggested, and none was required.
- False Positive Rate (FP): 22% - This is the percentage of observations of CCD Successes that are incorrect identified as impending CCD failures. These are cases where an intervention would have been suggested (leading to additional work within BT) but would have had no-effect since the case would have successfully completed anyway.
- False Negative Rate (FN): 43% - This is the percentage of observations of CCD Failures that are incorrectly identified as CCD Successes. These are the cases when an intervention within the process would have been advantageous but would not have been suggested.

The sensitivity of the classifier to True Positive cases whilst avoiding making too many False Positive predictions is the key balance to be struck. This could be turned into a

Table 4.6: Performance of the Immunos 99 classifier in terms of complete provisions rather than in terms of individual observations

Measure	Value
Number of instances	2545
Number of process failures	202
Number of false positives	1608
Number of failures with a fail detection	199

Cost/Benefit calculation by applying a nominal cost to each intervention and the cost of not intervening in terms of customer complaints or penalty payments for missing appointments and/or committed dates.

Classification Characteristics

The raw cross validation results show a partial picture of the results. The larger issue is not how the classifier behaves for individual observations but rather how it deals with whole orders. A lower true positive rate overall could be tolerated if more failures were detected during an orders lifetime (with the proviso that such a notification would need to be early enough for intervention to be possible). The timescales required for prediction to be useful, and the related level of acceptable latency for predictions is something that will require discussion with knowledgeable people inside the process.

In order to determine how the classifier is performing on whole orders the results from the initial classification were post-processed in order to count the number of complete orders, and capture the behaviour of the classifier throughout the observations of the order. The post-processed results are shown in Table 4.6.

To discuss the absolute frequencies first – there are 2545 orders in our sample of 100,000 observations of order touches. Of these, 202 failed their Customer Commitment Date (CCD). The classification performance shows that 1608 orders had at least 1 incorrect "CCD fail" prediction during their life. Therefore, had the system been live, BT would have needed to manually investigate over 70% of the orders. This level of false positives is untenable due to the high volume of orders for this product. More encouragingly there were failure detections in over 98% of orders that did fail.

Next, attention was turned to the time required in a process for a failure detection to be made. The median time to detect failure is an excellent 5-6 hours after order placement – this would likely give plenty of time for an intervention in the process to occur before the CCD.

An alternative measure of the amount of time since the start of the process is that of depth. This measures the number of touches between the start of the order and the current observation. The median depth for a successful failure prediction is 3, again this should be early enough for intervention, especially when a field engineering visit is required.

In summary, the true positive rate is very promising as there are only 3 cases that are never flagged as potential failures. However the prevalence of false positives is a problem – 63% of successful cases were flagged as failure at least once which is likely to be an unacceptable burden.

Investigating Characteristics of Failure Detections

The next investigative step was to examine the classified data to try and determine where the classifier was strong, and where it was weak. In this way assumptions about the data can be validated and suggestions about how the data and/or classifier might be adjusted to give improved performance can be formulated.

Accuracy by Depth

To look for suggestions of a relationship between the accuracy of the classifier and the depth of the current observation in the business process the values were plotted (Figure 4.4). The chart shows increasing depth on the x-axis and classification accuracy on the y-axis. Depth is defined as the number of tasks that occur up to and including the first task in the sequence included in the test data, for example the sequence $A \rightarrow B \rightarrow C$ is at depth 0 if these are the first 3 tasks in the business process. The more erratic behaviour of the line is coincident with the reduction in the number of observations at those depths. These points should be given less credence in analysis.

It is interesting to note the increase in accuracy between depths 9 to 20. This suggests that the middle of the process provides the best opportunities for prediction. To determine if this is reasonable would require knowing what the tasks were at each depth, however there is no direct relationship between depth and task.

True Positive and False Positive Rate by Depth

The next test examined the relationship between depth of observation and the two most important classifier metrics, True Positive and False Positive rates (Figure 4.5).

The chart suggests that true positive rate trends downwards over time. This suggests

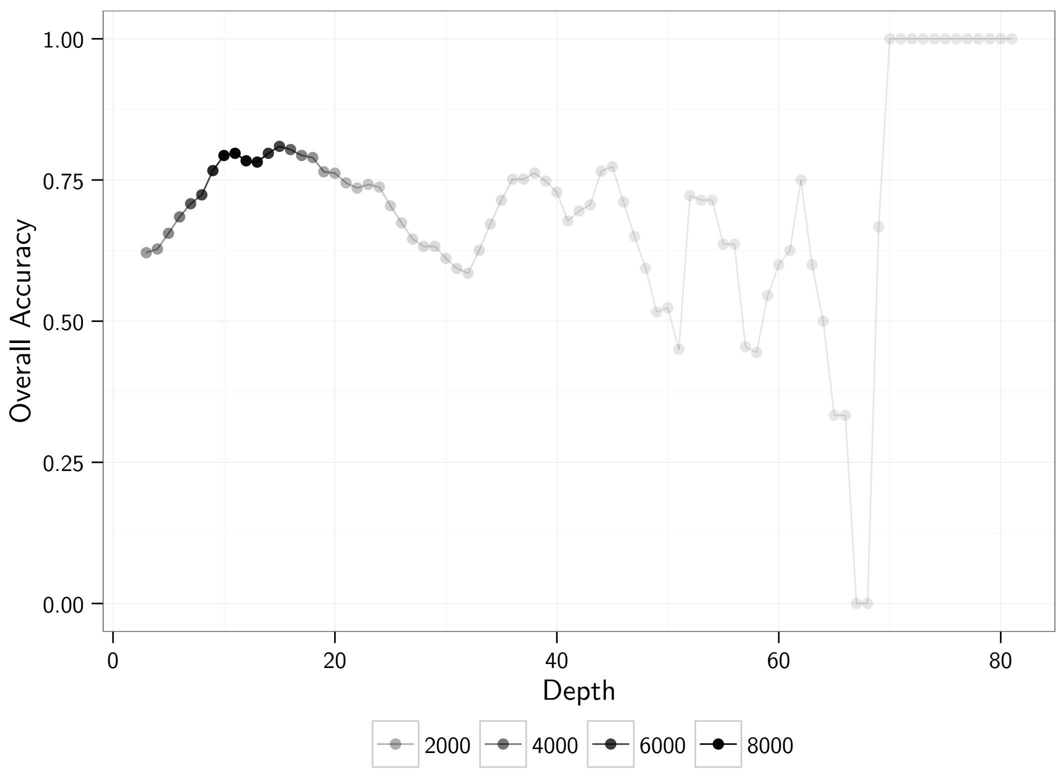

Figure 4.4: Accuracy of the Immunos 99 classifier by depth. Points indicate the accuracy at each depth, the opacity of the points represents the number of observations summarised at that point.

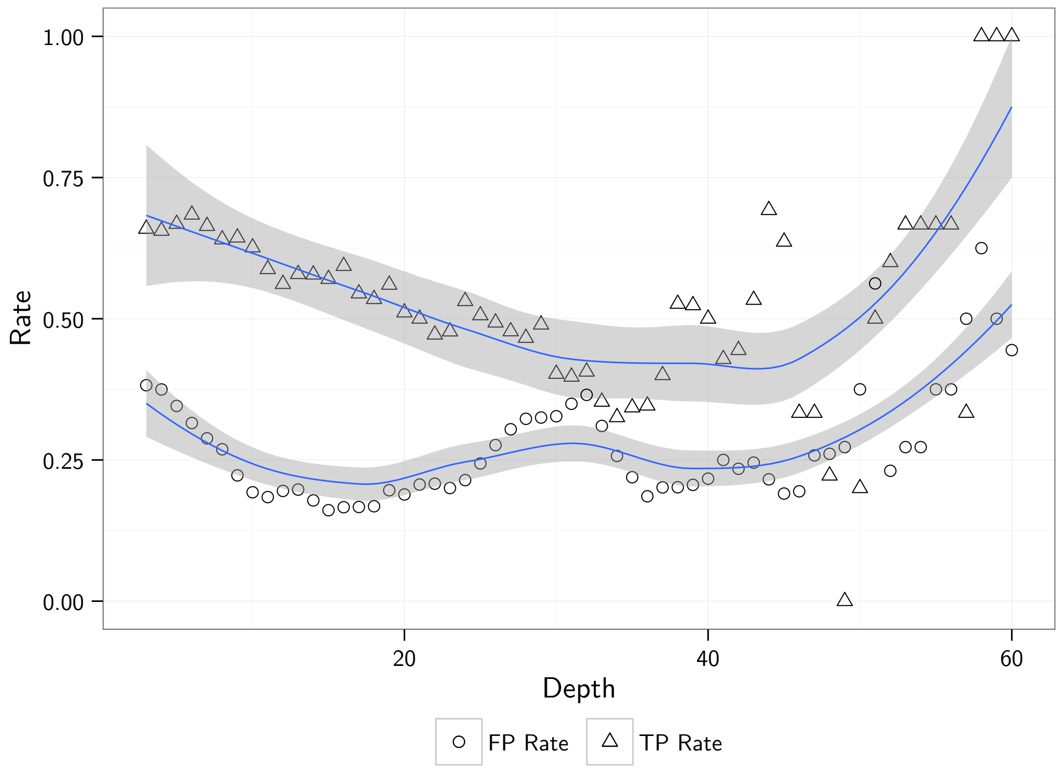

Figure 4.5: True positive and false positive rates plotted against depth. Confidence intervals are drawn at the 95% level.

the classifier might be leaning towards marking the cases as failures early on and then correcting that incorrect prediction as the case continues. The spike in false positive rate between depths 25 and 53 is an interesting anomaly that would require additional work with people involved in the process execution to understand.

True Positive and False Positive Rate by Task

Whilst the depth of the process gives us some idea of performance over time it does not give useful information about the actual tasks that were performed because the tasks at each depth are not the same in each case. Figure 4.6 shows the true positive rate on the x-axis and the false positive rate on the y-axis of a scatter chart. The tasks with the greatest predictive utility are those with high true positive and low false positive rates. Actual task names are replaced with generated identifiers to mask their identity.

We can deduce a set of tasks with desirable classification characteristics in this case. Further investigation into the suitability and appropriateness of the higher performing tasks required discussion with the process owners.

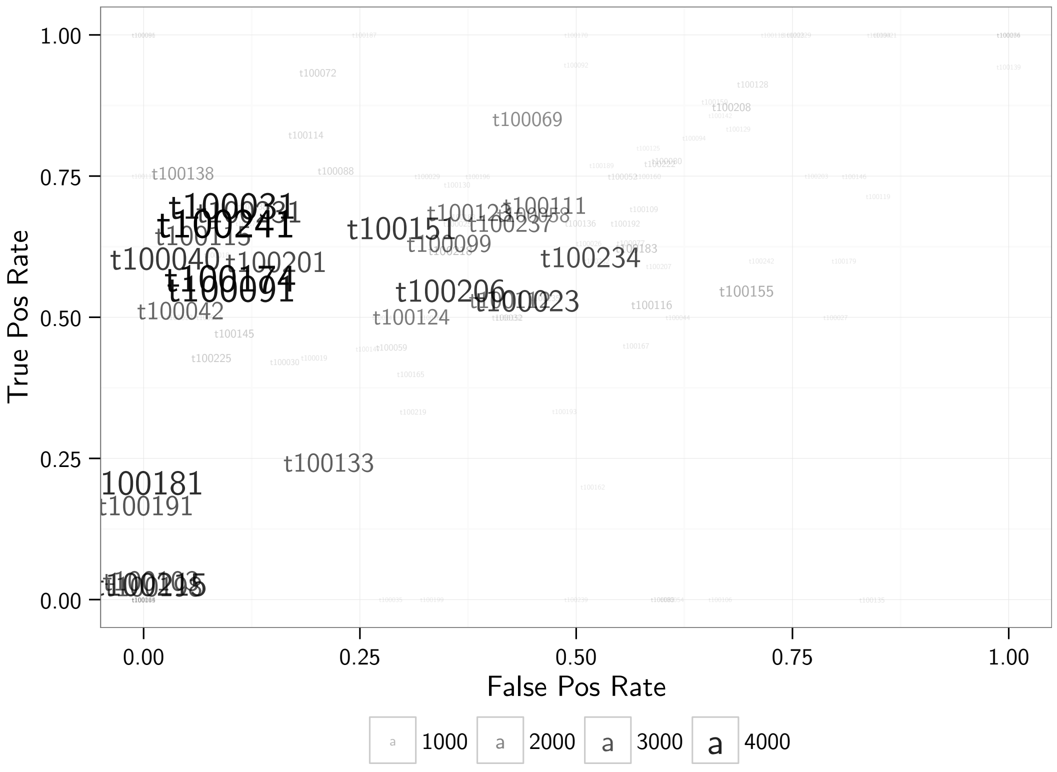

Figure 4.6: Scatter plot showing the true (TP) and false positive (FP) rates for each task with FP on the x-axis and TP on the y-axis. The values for each task are at the centre of the task identifier. Opacity and size indicate the number of observations for each task. Tasks with higher predictive utility would appear in the upper left of the chart.

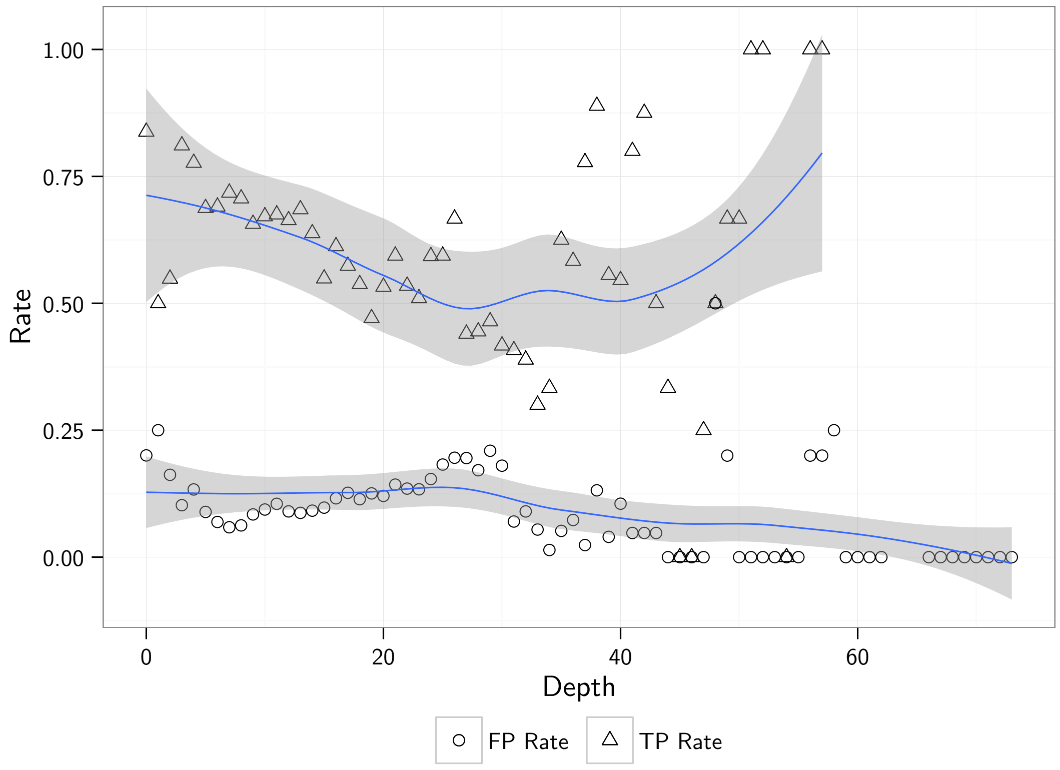

Figure 4.7: Classification performance using best performing tasks. Confidence intervals are drawn at the 95% level.

Improved Performance using Best Performing Tasks

The next test was to determine whether the performance of the classifier improves if only the best tasks were used for the classification. The tasks used where those 14 that had TP > 0.5 and FP < 0.25.

Figure 4.7 shows the accuracy by depth of the chosen tasks in terms of FP and TP rates. The same trend downwards over time can be observed in the chart, but an improvement on the previous tests can be observed in the reduction in false positive rate with increasing depth.

An interesting result from Figure 4.7 is the increase in TP near the start and end. There could be a number of potential reasons, with the following hypotheses proposed:

- At lower depths there is more training data because some process executions contain fewer tasks than others. Process executions containing less tasks will contribute observations at earlier depths and not at higher depths.
- Cases with extreme depths are almost certainly to have taken so long that the process has already missed the CCD. This would be very easy for the classifier to detect and use.

Classification Performance Against Clock Time

The notion of depth in the process data is an abstraction of time. Some processes will run for over 30 days and others for less than 4 yet they may contain a similar number of events and therefore similar depths. Business performance of the classifier is intrinsically tied to how much real time is available to prevent a failure from occurring. Identifying failures only 10 minutes before they occur is not a useful outcome. Figure 4.8 shows the classification accuracy in terms of true positive and false positive rates with the x-axis representing real time elapsed since the start of the process at the point of observation.

Examining the points in the chart shows a decrease in false positive rate after 8 days. This is interesting, and likely related to the failure of an initial engineering activity, however our most useful interventions would be in that timescale so this is of concern.

Classification Performance Against Time to CCD

Examining the performance with increasing time is interesting, but given the product in question is provisioned with reference to a Customer Commit Date (CCD) it would be interesting to examine how the classifier performance changes as the CCD approaches and passes. A key question is can the system intervene accurately and with a sufficient number of days before CCD to make a difference to the customer. Figure 4.9 shows the TP and FP plotted on the y-axis against time to CCD on the x-axis.

The most interesting observations to come out of this chart is that it shows the classifier has 0 false positives after CCD, which are all failures by definition. It is interesting to see that Immunos 99 has picked up that fact in the classifier training phase. Another interesting feature is the bump in TP rate about 10 days before the CCD. Root cause analysis to determine why the bump in performance might exist be will be performed later with the process owners. The highly erratic nature of this graph makes conclusions drawn from the trend lines suspect.

Including Frequencies

The sporadic nature of the results shown in Figure 4.9 combined with the wide confidence intervals on the best fit lines suggests that performance is not steady. This might stem from uneven distribution of data over time to CCD. Therefore the previous plot was reproduced as Figure 4.10 but this time with point size representing the number of observations behind that point.

Figure 4.10 clearly shows the skew in the data towards events occurring on the day. This

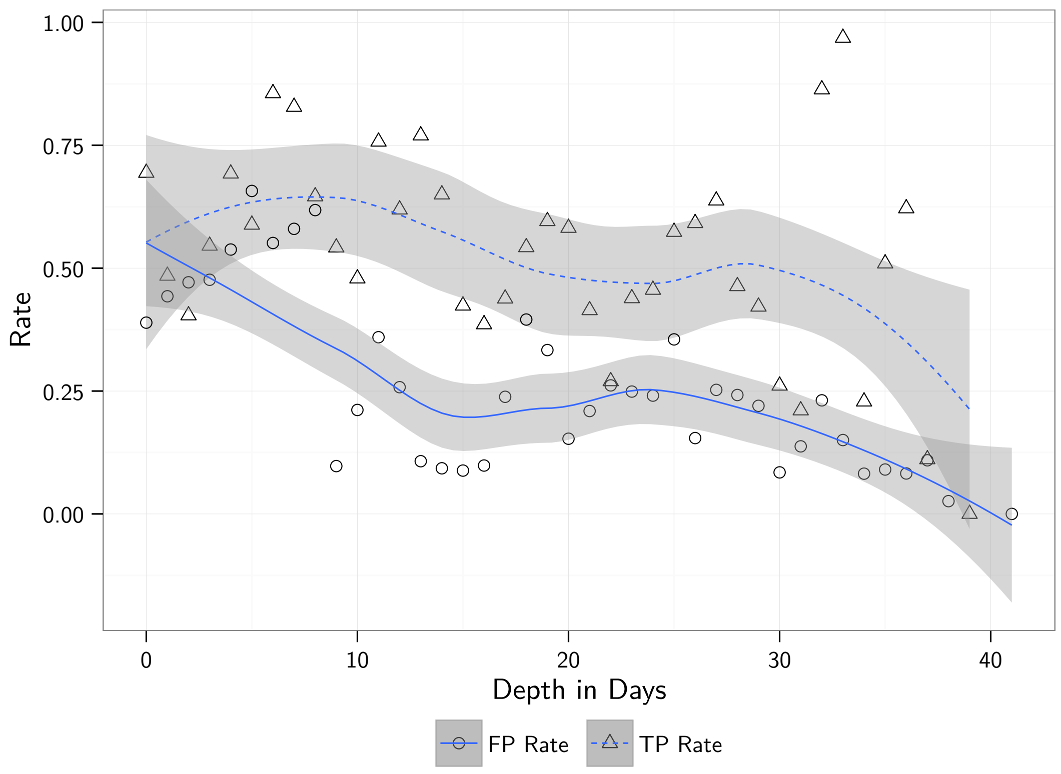

Figure 4.8: Classification Performance plotted against clock time. Confidence intervals are drawn at the 95% level.

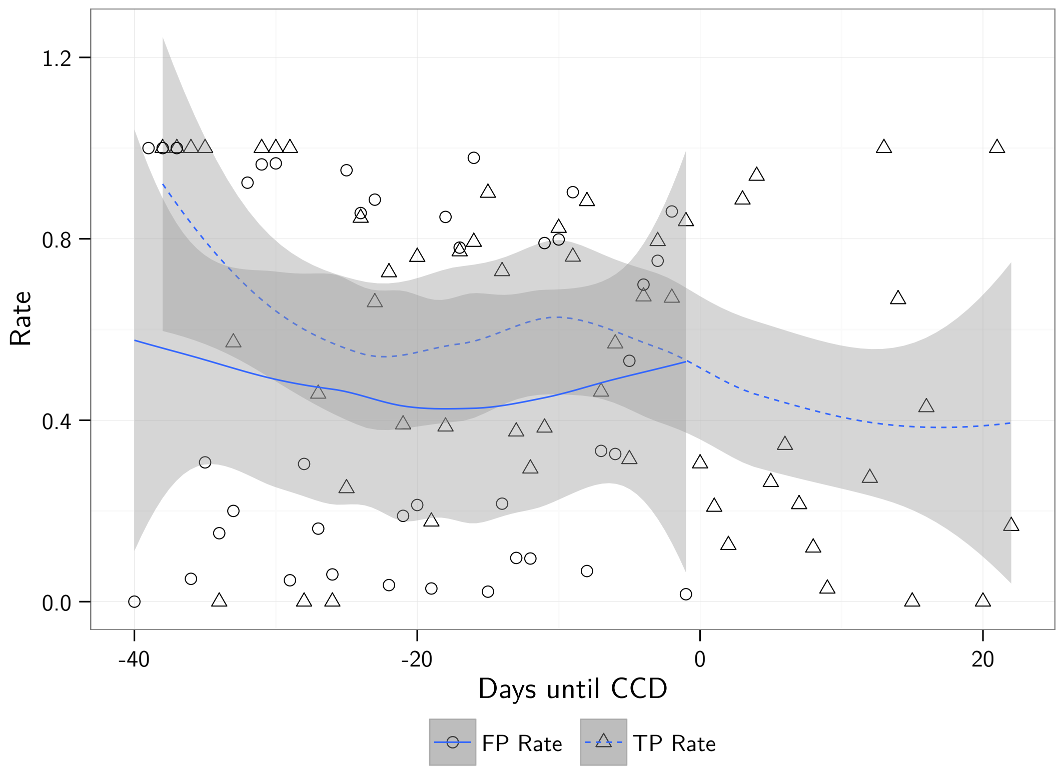

Figure 4.9: True positive rate and false positive rate plotted against time to CCD. Confidence intervals are drawn at the 95% level.

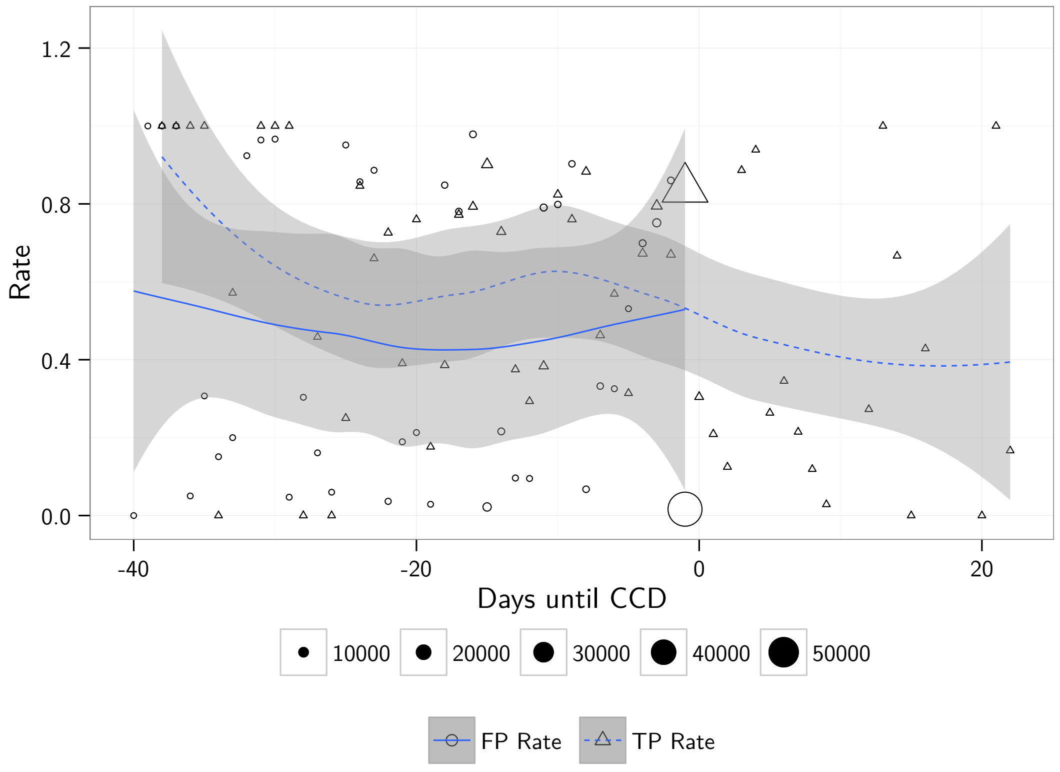

Figure 4.10: True positive rate and false positive rate plotted against time to CCD including observation frequencies. Confidence intervals are drawn at the 95% level.

is problem for training the classifier on the earlier data, and also for intervention. Even correctly predicting a failure on the day does not leave sufficient time to resolve the problem.

Future Work

There are a number of possible actions that could be taken to attempt to increase accuracy, these include using a less detailed data set, although that has been briefly tested and does not appear to have a significant effect. Another option would be to adjust the training data provided to the classifier to only include those observations from the start of the process out to the 'point of no return' where no successful intervention is possible. This may help the classifier focus of the more important parts of the process more accurately. This work is scheduled to take place within BT's 2015/2016 strategic research programme.

4.7 Conclusions

This chapter examines four immune-inspired classification algorithms in terms of their ability to predict the outcome of business process executions from trace data gathered on those executions. Of the algorithms tested only Immunos 99 shows performance characteristics worth pursuing in future work. The current performance of the Immunos 99 classification algorithm is promising although there is work to do to reduce the false positive rate sufficiently for business use.

Following the initial testing which used the same test data as used for the classical classification techniques in Chapter 3, results of a internal BT case study examining the strengths and weaknesses of Immunos 99 on a completely new data set are summarised. This reiterated the findings of the initial testing that the false-positive rate exhibited by Immunos 99 is the key issue that would need to be resolved before the algorithm could be considered the ideal choice for business process data.

At this point both classical and immune inspired classification techniques have been tested against business process data. Each of the two types of techniques has drawbacks. The decision tree techniques require a human to spend time tuning cost parameters when the algorithms are trained. The requirement for human tuning will increase the costs of implementing that technique to take it out of the reach of low volume processes. Immunos 99 however shows the opposite: in high volume processes the number of false positives is likely to make the Immunos 99's performance untenable, however these are exactly the processes which would be able to afford tuning of the decision tree/MetaCost solution.

In combination, Chapter 3 and 4 have shown the it is possible to make predictions on business process outcomes using classifiers derived from historical execution data. These classifiers are the first stage of creating a self-healing business process system as they will allow the system to determine where intervention would be required. The next task is to determine what interventions can be made and which of those interventions is most appropriate. Chapter 5 discusses the determination of recovery strategies and their selection in detail.

Chapter 5

Identifying & Applying Recovery Strategies

5.1 Introduction

In Chapters 3 and 4 techniques to predict the outcome of a business process execution have been discussed. Being able to predict the ultimate success or failure of a business process execution is not sufficient for a self-optimising business process system to be complete. In addition to predicting when a business process might fail the system must also be able to determine and then execute an appropriate recovery strategy to try and ensure the process completes successfully in terms of the identified business goals.

A recovery strategy can take one of two forms. The first form would be an sequence of task executions that is predicted to return the executing business to a state where the predictive element of the system begins to predict success. At the end of the recovery path the system resumes executing the process as originally intended.

The second form of recovery strategy would be a longer sequence of states that would map out the entire remaining lifetime of the business process execution from the point of failure being predicted to the eventual completion of the business process. This would mean that once the classifier has predicted a failure, the process will never return to normal control.

In small, well understood systems, it might be possible for an expert user to determine in advance all of the possible failure states of a business process and to determine the best path to success in each of those failure states. However in complex business processes such as those discussed in this thesis such complete knowledge is not usually possible.

Table 5.1: Example of an historical recovery with classifier data using a sliding window of size 3

#	Task 1	Task 2	Task 3	...	Real Outcome	Classifier Outcome
1	A	B	C	...	Success	Success
2	B	C	D	...	Success	Success
3	C	D	E	...	Success	Failure
4	D	E	F	...	Success	Failure
5	E	F	G	...	Success	Failure
6	F	G	H	...	Success	Success
7	G	H	I	...	Success	Success

An alternative solution that does not require a human to predetermine recovery scenarios is then to be preferred. Given that a well documented and machine understandable process design is not available, as discussed in previous chapters, solutions that allow the self-optimising process system to learn recovery strategies from historical data, and that allow the system to determine which recovery is best suited to unfamiliar scenarios are ideal.

This following section discusses in detail one approach to extract recovery strategies from historical execution data using a predictive classifier such as those discussed in Chapters 3 and 4. Discussion of how to best determine which recovery to apply in a given failure scenario is presented in Section 5.3.

5.2 Determining Recovery Strategies

A process recovery could be defined as the the return of a process execution to a state in which success is predicted, leaving the remainder of the process execution to standard workflow mechanisms, this is referred to as a short-type recovery. The alternative long-type recovery strategy is defined as the sequence of tasks required to complete the current process execution in a successful state, taking a process execution out of the normal flow entirely once failure has been predicted.

A classifier that has been trained for use in process prediction could be applied to a set of historical process data, which could be the same or different to that used for training, in order to discover recovery strategies. When the classifier is applied to historical data, the training data is extended with an additional attribute showing the outcome of classification at that stage of the process. Using this additional column the data set can be processed to extract examples of process executions where a recovery has been observed.

Table 5.1 shows an example extract from historical data that has been evaluated by a classifier. Row 3 is the first row of the execution that shows a failure prediction, this is the point that the recovery starts. Row 6 is the next row that has a success prediction, therefore our short-type recovery strategy consists of the transitions used on rows 3-6, i.e. $E \rightarrow F \rightarrow G \rightarrow H$ and the long-type recovery strategy consists of the transitions from row 3 to the end: $E \rightarrow F \rightarrow G \rightarrow H \rightarrow I$.

When this approach to generating recovery strategies is applied to a large training set this produces a database of observed recoveries which can be further augmented with reliability metrics. The complete listing of data recorded for each recovery is:

- The state of the process immediately prior to the recovery beginning, consisting of:
  - The attributes of the process execution
  - The task history of the process execution (e.g. $A \rightarrow B \rightarrow C$)
    - The temporal state of each task
    - The name of each task
    - The position of the task within the process execution
- The short recovery path
- The long recovery path
- Number of times the recovery is observed
- Number of times the recovery results in a successful outcome
- Number of times the recovery results in an unsuccessful outcome.

Depending upon the business's appetite for risk, at this stage rows with an observed number of successful interventions or a percentage success rate below a threshold may be removed from the database. Removing entries from the recovery database will prevent them from being considered for automated recovery.

5.2.1 Effectiveness of Strategy to Discover Recoveries

The approach to discovering recovery strategies described above was applied to the data sets described in Section 3.1. This produced a table for each data set of discovered recoveries and some basic performance metrics for those recoveries. Summary details are shown in Tables 5.2 and 5.3.

Table 5.2: Statistical summary of discovered recovery strategies, using short paths

Data Set	Number of Recoveries	Mean Obs. per Recovery Path	Mean Success %	Variance of Success %
10-task	144	182.75	85.17%	501.00
Full	3835	1571.23	43.08%	618.09

Table 5.3: Statistical summary of discovered recovery strategies, using long paths

Data Set	Number of Recoveries	Mean Obs. per Recovery Path	Mean Success %	Variance of Success %
10-task	144	163.32	84.93%	539.63
Full	3835	1188.84	40.47%	753.48

From Tables 5.2 and 5.3 it can be seen that the longer paths are observed less in the historical data. Longer paths are likely to be observed less because of the variation and noise in the training data. The longer a path is the less likely it is to occur by chance.

More interesting however is the difference in success between the simplified 10-task data set and the original data in the full data set. The drastic reduction in performance between these two suggests that the more complex and chaotic a data set the less successful the recovery is. That simpler processes lead to more consistent and correct outcomes would seem to be intuitively correct. Further work to determine how complexity correlates to the number and quality of recoveries produces for a business process would be an excellent candidate for further research.

5.3 Implementing Recoveries Automatically

To self-optimise a business process and ensure that it completes successfully requires three things. First, the ability to detect potential issues (Chapters 3 and 4). Second, options for corrective action to to be taken as discussed in the first part of this chapter and third, the application of those recovery strategies to the running business process. The remainder of the chapter discusses how recovery strategies can be used in an adaptive process system that could support the execution of business processes on existing infrastructure. This chapter also discusses how to best select a recovery strategy for a particular failure prediction, and how the predictions might be adjusted over time to take into account changes in the process and environment.

In order for the proposed process system to be considered self-optimising/self-healing there should be some feedback and learning from completed business processes. The simplest way to incorporate feedback mechanisms is by allowing the results of completed processes to influence the selection of recovery strategies for those business processes that are predicted to fail.

In order to incorporate these features into an existing business process system, a companion system would be required that would monitor the business process and intervene where required. The autonomic business process system will need to interface with existing workflow systems to report the progress of business processes, detect potential failures, and adjust workflow to optimise for success.

The remainder of this section introduces a proposed a proposed self-optimising process system incorporating feedback mechanisms, which can be used to augment existing workflow systems with self-healing properties for individual process executions. A block diagram showing the data flows between components in the combined system is shown in Figure 5.1 and a detailed description of the steps taken by the self-healing system to influence the existing workflow immediately follows. The proposed system was derived with reference to what would be acceptable to the industrial partner.

Each time a business process is updated in the controlling workflow system a notification is provided to the autonomic process system. Each update flows through the proposed autonomic system as follows (with reference to Figure 5.1):

- Workflow system notifies the system of an update to a monitored execution
- If the workflow is not yet complete then:
  - The Classifier/Monitor module makes a prediction for the closing status of the execution based on the current state
  - If the prediction is success then no further action is taken
  - If the prediction is failure then the execution is passed to the Recovery Engine which will consult the Recovery Database for the best matching recovery (see Section 5.3.2).
  - If a suitable recovery cannot be found, then the execution is passed to a Human Expert with the best candidate. The Human Expert will accept/edit the suggested recovery and pass it back to the Recovery Engine.
  - The recovery workflow is then returned to the Workflow System for execution
- If the workflow is complete then:
  - The Classifier/Monitor passes the update to the Learning/Adaptation (L/A) module
  - The L/A module uses the final outcome of the execution to update the Recovery Database with details of this execution (success and total counts)

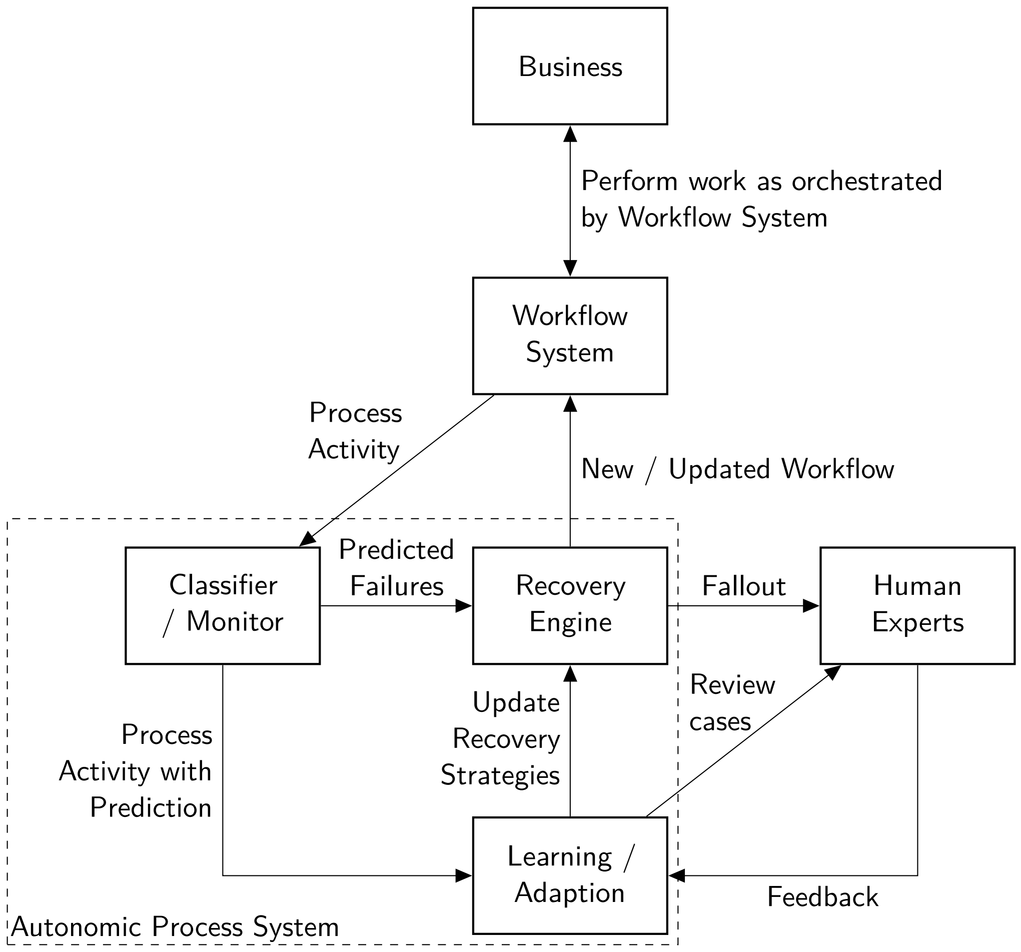

Figure 5.1: An outline of an autonomic process system

- If the execution contained an intervention, the L/A will forward the details of the process to the Human Experts. In high volume processes this will be done with an appropriate proportion of the cases as the entire volume is likely to be untenable. The proportion of cases to be sent to review is a system parameter to be agreed with the business.
- The expert will validate the workflow and by indicating which of the recovery steps were required, which will score the recovery in terms of efficiency. The efficiency score will be used by the Recovery Engine to adjust a weight attached to each recovery. This weight would be included in the scoring mechanism used by the Recovery Engine to select the most appropriate recovery.

Figure 5.1 shows the inclusion of human experts into the autonomic process system. Human experts in this context are people with an in-depth knowledge and understanding of the process and the underlying systems. These people will be able to take a problematic process execution and resolve any issues that are impacting its execution. The experts will also be able to examine the recoveries stored in the system and rank them based on their expected efficiency and effectiveness using their inherent knowledge of the process.

The human experts therefore provide three benefits to the system. First, the system will be able to learn from the knowledge of the humans within it by incorporating human feedback on the suitability of chosen recoveries. Second, to validate the interventions made by the system. Third, to assist in scoring the recovery strategies for use in the adaptation of the recovery database. By utilising human expertise in this way the system can take over the management of the common simple issues using the automated intervention techniques proposed and allow the humans to concentrate on those cases that are best suited to human problem solving.

The feedback flows described above and indicated in Figure 5.1 would be implemented as a weight (or weights) assigned to each recovery strategy. The current weight of a recovery strategy will be incorporated in the calculation used to determine the most suitable recovery for the current process state. When a recovery intervention was determined to have been successful, the weight of the implemented recovery would be increased by a small amount, and vice versa. The human experts in the system should also influence the weights through their review process. The experts score the selected recovery based upon the number of tasks in the recovery path that are absolutely necessary for successful recovery. The proportion of absolutely necessary tasks to the number of tasks in the whole recovery path would represent the estimated efficiency of the recovery. The efficiency metric would be used to adjust the weight of the recovery strategy with more efficient recoveries having their weights increased.

It is likely that the process being monitored will evolve over time. Process evolution can occur due to changes in the process design, changes to working conditions such as supporting system changes, personnel changes, or changes in the external environment (e.g. extreme weather events impacting field engineering activities [2]). Another addition to the scoring mechanism would be an atrophy parameter. This would reduce the weighting of all recoveries by a small amount each time period, causing those recoveries which are no longer being selected to be less likely to be selected for use. The changes in weights on the recovery scenarios would become useful input to business process improvement programmes.

The system described relies heavily upon the use of weights to allow the system to adapt to its business process and environment. The use of weights has a number of issues that need to be addressed. The first of these is to determine appropriate values for the initial weights, and second to determine the appropriate changes to be made to the weights based on the feedback flows in the system. The investigation of the first of these problems is discussed beginning in Section 5.3.1. The investigation of the second part would be a prime candidate for further work.

The proposed approach feels intuitively useful because it wraps existing business process infrastructure enabling easier integration with those systems. There are a number of barriers to testing the proposed system in a real deployment. First, the source data that is available for analysis relates to large scale, mission critical process for the business. Using a large scale process means there is a large volume of data available for analysis, however due to the mission critical nature of the business process changes to its infrastructure are not made lightly. Second, the human elements of the process system are practically impossible to include in a test situation. Whilst not all parts of the proposed system can be analysed in full detail, the selection of recoveries is analysed further in the next section.

5.3.1 Choosing a Recovery

The proposed system discussed in Section 5.3 and illustrated in Figure 5.1 selects the most appropriate recovery from the recovery database by finding the recovery strategy that has occurred at the process state most similar to the current process state. A key element in selecting the recovery most appropriate for the current situation is the notion of similarity. Intuitively, the recovery most similar to the current process state should be chosen for automatic implementation. This is subject to the recovery being retained in the recovery database, Section 5.2 discusses potential pruning of the recovery database based on business needs (e.g. removing recoveries with low success rates).

There are a number of features which could contribute to a similarity score for a given particular process state (S) and recovery strategy (R). Data for each of these elements is recorded for the recoveries in the recovery database as discussed in Section 5.2. These include:

- Exact Path Similarity;
- Partial Path Similarity (also referred to as Task in Path);
- Attribute similarity;
- Temporal similarity.

For Exact Path Similarity ($\text{ExactPathScore}(S, R)$), the sequence of tasks executed in the current process state is tested to determine if it matches exactly the sequence of tasks executed immediately prior to the recovery path. The comparison is a test for equivalence between the string constructed of the 3 tasks in the path for the current process state and recovery. The test returns 1 when the path matches exactly and 0 otherwise. For example the path $A \rightarrow B \rightarrow C \overset{?}{=} X \rightarrow Y \rightarrow Z$ would result in a 0 score as the strings are different.

Partial Path Similarity ($\text{PartialPathScore}(S, R)$) is calculated on the degree of match between the tasks in the process state at the start of the recovery and the current process state, irrespective of the relative position of the task in the sequences. The function returns the number of tasks in the recent history of S that match the equivalent in R. When a task has been matched, it is removed from matching consideration for the remaining elements in the sequence. For example, given a recovery with the path $A \rightarrow B \rightarrow C$, a process state with path $B \rightarrow B \rightarrow C$ would match 2 of the path elements, B and C. Similarly the path $A \rightarrow A \rightarrow A$ and the recovery $A \rightarrow B \rightarrow C$ would match 1 element, A.

Attribute Similarity is calculated independently for each attribute common to the process and recovery. The total attribute similarity is calculated by multiplying the number of matching attributes with the attribute match score. Attributes are considered to be matches if and only if they are exactly equal. $\text{AttributeScore}(S, R)$ returns the number of attributes in S that match those in R. For example, Table 5.4 shows a process (column 2) and recovery (column 3) which share 4 attributes (one per row). The last column in Table 5.4 indicates which attributes are exact matches between the Process and Recovery. The total row shows that 2 of the 4 attributes match. Therefore in this case $\text{AttributeScore}(S, R) = 2 \times \text{AttributeWeight}$. The tasks in the execution path and the exact path itself are considered in ExactPathScore and PartialPathScore respectively and are not considered as attributes for the calculation of AttributeScore.

Temporal Similarity does have some complications since the processes are unlikely to feature identical timestamps, instead the temporal behaviour may exhibit periodicity over

Table 5.4: Example attributes for a recovery and a process state

Attribute Name	Process Value	Recovery Value	Matches
Customer	Customer S	Customer T	0
Product	PSTN	PSTN	1
Include Broadband	Y	N	0
Region	Cumbria	Cumbria	1
Total Matches:			2

$$\begin{aligned} Score(S,R) = {} & \text{ExactPathWeight} \times \text{ExactPathScore}(S,R) \\ & + \text{PartialPathWeight} \times \text{PartialPathScore}(S,R) \\ & + \text{AttributeWeight} \times \text{AttributeScore}(S,R) \end{aligned} \tag{5.1}$$

a number of timescales. Such periodicity would be useful knowledge for a predictive algorithm. For example, process failures may be more likely in processes or tasks starting on a Friday or starting after 3:00pm. To attempt to overcome this issue the start and end timestamps for each task in the task sequences are decomposed as discussed in Section 3.2.1. Decomposing the temporal elements does not mean these are omitted, instead the attributes that are created through decomposition are incorporated into the standard attribute scoring process described above. For example, if the start time of S and the start time of R are both PM times, then this would count as an attribute match in the same way as if the customer was matched.

Each of the listed features is available in the data sets that have been used throughout this thesis. The accuracy of each attribute is dependent on the source data, the quality of which can be variable, as discussed in Section 3.1.1.

The overall similarity score $Score(S, R)$ incorporates all of the above elements as shown in Formula 5.1. There is a partial dependency between the exact path match and the partial patch match, however this is simply a way of further rewarding exact matches over those matches that contain the same tasks but in a different sequence.

5.3.2 Scoring a Recovery using Similarity Measures

To calculate the similarity score for a process state, S, and recovery, R, each similarity measure is calculated and the results are summed together to give the overall similarity score for S and R (see Formula 5.1). The recovery that will be selected for a given process state is the one with the highest similarity score.

To test the appropriateness of these similarity measures the recovery strategies determined in Section 5.2 were used. Each predicted failure state from the 3-step training data set and the 10-Task simplified variant of that data set (see Section 3.2.1) were scored using Formula 5.1 and the initial weights for each match type shown in Table 5.5. These weights, used for initial testing, are not scientifically determined, instead they are chosen so that the resulting values can easily be decomposed to the different match types by visual inspection and charting. For a real deployment, these weights would likely need to be tuned or trained to give acceptable real-world performance.

The distribution of the highest similarity score for each process state and recovery is shown in the histograms in Figure 5.2. For the 10-Task, 3-Step dataset the histogram shows there were a negligible number of matches on attributes only (which would score between 0-100) and the most matches are on full paths (in the 1000+ bracket). Similarly, in the Full Dataset there are few matches on attributes only and again the most matches are on full paths. This is interesting because it shows that most of the observed paths in the dataset do have at least one instance where a recovery has been observed. Knowing that virtually all of the observed paths have a recovery is positive in that it suggests that this means there is a detectable path to successful completion for most process execution issues.

To determine realistic weights appropriate to a business process in the real world some parameter optimisation would be required. A typical approach to optimise a set of parameters for a given situation is to use an evolutionary algorithm and a fitness function which would be used to map the proposed weights to a result that describes the suitability of that set of input parameters (e.g. [122]). When working with process data however a fitness function can be very difficult, if not impossible, to define because the success of the executing process is ultimately dependent on the external environment and upon factors that are not under the control of the business process system. Some of these environmental failures could be factored into a fitness function however, these are highly process specific and non-trivial to define and reason about. An example of an environmental failure would be the situation in which an engineering visit is required and the customer is not at home when the engineer arrives for the appointment. The process may be considered a failure because the fault is not resolved even though the business process was flawlessly executed. The difficultly of defining a fitness function in an analogous multi-agent simulation is discussed in [123].

Table 5.5: Weights used for initial testing

Element	Weight
ExactPathWeight	1000
PartialPathWeight	100
AttributeWeight	1

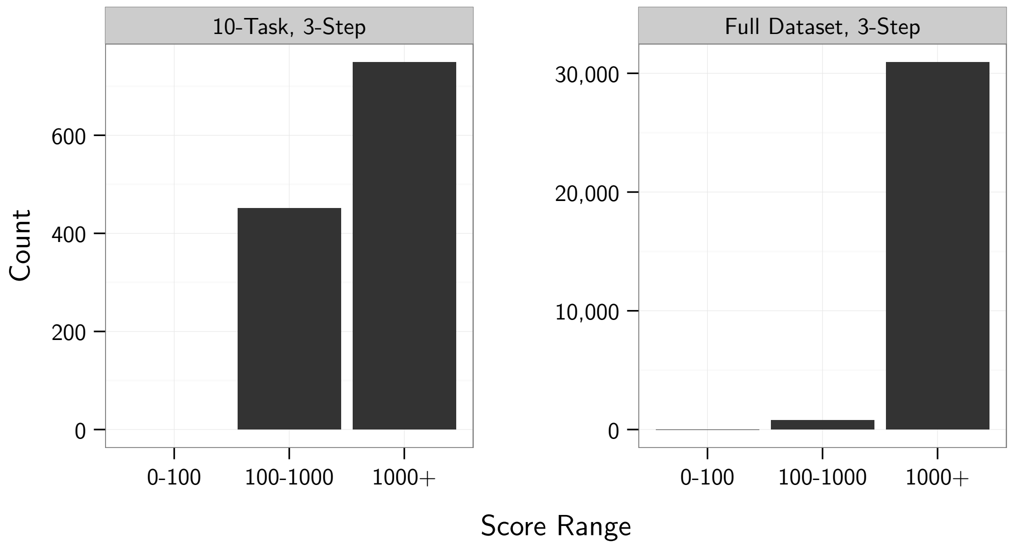

Figure 5.2: Histogram showing distribution of maximum similarity scores for recovery/process state comparison in the test data sets using the scores from Table 5.5.

As a fitness function is not practical, given the difficulties discussed, an alternative method to gauge the suitability of a set of scoring criteria is required. The data sets that are available have a large amount of historical data for the process being experimented on. Using historical data to validate the outcome of simulations is discussed by Sargent in [124], the approach is to partition the data set into separate sets used for building and validating the simulation. To determine recovery suitability, historical data is leveraged to replace a fitness function by counting the number of successful processes exhibiting the recovery behaviour and dividing by the total number of processes exhibiting the behaviour. This allows analysis of different scoring options based on the historical success of the chosen recoveries.

Details of the success rates of the selected recoveries for the weights in Table 5.5 are shown in Table 5.6 for both the simplified 10-Task and Full Data sets. The table shows the proportion of selected recoveries that have been observed in the historical process data and the median success percentages of the historical processes that contained the selected recovery path.

It is immediately apparent from the table that the simplified data appears to be significantly more amenable to the described recovery approach with its median success rate of 83.77% than the more complex and noisy full data set with the far lower median success rate of 11.11%. This is an interesting and important point. It suggests that simpler processes (or simplified data from complex processes, perhaps at a higher conceptual level, see [1]) produces higher quality recoveries. This may be due to the fact that the simplified data contained fewer tasks and therefore less noise and variability in the task sequences.

Table 5.6: Success percentages for historical process data containing the selected recovery path for the selected recoveries using the weights from Table 5.5

Data Set	Recos. Observed	Mean Success %	Median Success %
10 Task, 3-Step	67.00%	74.32%	83.77%
Full Data set, 3-Step	97.00%	28.21%	11.11%

Such a reduction in variability could easily account for the higher proportion of recoveries that were observed in historical data for the 10 Task, 3-Step data set. There are possible approaches that could be used to reduce the variability of the paths in the full data set. For example, paths that occur infrequently could be exempted from the automated recovery process, allowing humans to take the complex cases and leveraging the automated approaches for common issues. Work is ongoing within the industrial partner to determine how a complex process should best be represented to get the best value out of an automated recovery approach.

5.3.3 Realistic Scoring

Whilst the weights described in Table 5.5 were appropriate for initial testing, in a realistic process system the weights associated with the different match types would need to be tuned for best effect. In order to determine how best to approach this, it was necessary to test different weights and attempt to discover how the different weights impact the quality of selected recoveries. The data set used for the testing of the different weights was the full data set. The full set was chosen because its initially poor results had the largest scope for improvement (see Table 5.6). If a significant increase in the quality of the recoveries was observed for certain testing weights, then it would have revealed that careful tuning of these parameters would be required in a real system. Alternatively, if the analysis revealed no significant chance in behaviour for the different sets of input parameters, then it is likely that it was the underlying complexity of the data set and business process that had the largest impact on the quality of the chosen recoveries.

The relative importance of the different weights to be applied to the different match types were unknown. In order to try and gain additional insight into the importance and configuration of the match type weightings a sensitivity analysis was performed. This analysis tested for correlations between the different weights and the quality of the produced recoveries, as measured by the previously discussed metrics of success count and success proportion.

To perform the sensitivity analysis the Spartan [125] Toolkit's Latin Hypercube analysis option was chosen because it was able to test the effect of changes in all 3 of the weights

(ExactPathWeight, PartialPathWeight & AttributeWeight) simultaneously, revealing combinatorial effects as well as individual impacts of weights on the quality of the selected recoveries. First, Spartan generated 100 sets of potential scores for the 3 different match types. Each score would be varied over the range 0 to 100. The 0-1000 range used for the initial testing was not used here because the generated parameter values will not produce results that can be decomposed by visual inspection. Each test run would consist of a complete scoring of the Full 3-Step data set. The exact weights used for each match type in each test are shown in Appendix B.

Spartan requires one or more numerical inputs representing the quality of the produced recoveries for each parameter set. Each run through the historical process data using the identified recoveries would generate one result per observation that was a predicted failure. To summarise the performance of each set of weights over the entire data set, the median of 3 output variables were recorded for each set of weights. Median was chosen over mean because the distribution of the recovery quality metrics is unknown and cannot be assumed to be normally distributed. The recovery quality metrics that were tested for sensitivity were:

- The median number of historical observations of the selected recovery that were successful;
- The median number of historical observations of the selected recovery overall;
- The median percentage of historical observations of the recovery that were successful.

5.3.4 Sensitivity Analysis Results

The sensitivity analysis produces a correlation coefficient for each input variable. Results for each output are shown in Tables 5.7, 5.8 and 5.9. All figures in those tables are rounded to 4 significant figures. Scatter plots showing each combination of input and output values are shown in Figures 5.3 through 5.11.

The results of the Spartan analysis of the proposed scoring scheme show no statistically significant relationships between any of the weights and the success of the selected recoveries at the correlation coefficient > 0.9 level. These results show that the scoring weights are not a significant driver for recovery quality.

The shapes of the charts also show that the output values that are counts are always in the range $[0, 2]$, this shows that there is little consistency in the business process since each recovery is only observed few times in the historical process data. The lack of significant correlations and the low frequency of each specific process paths suggest that it is likely that the underlying complexity of the business process being examined is the key factor behind the low quality of the selected recoveries.

Table 5.7: Table showing correlation between input weights and the number of matching recoveries in the historical data

Score Type	Correlation Coefficient	P-Value
Exact Path Weight	0.4223	0
Partial Path Weight	−0.6793	0
Attribute Weight	0.4537	0

Table 5.8: Table showing correlation between input weights and the number of matching successful recoveries in the historical data

Score Type	Correlation Coefficient	P-Value
Exact Path Weight	0.1709	0.089 32
Partial Path Weight	−0.1742	0.083 00
Attribute Weight	−0.2373	0.016 68

Table 5.9: Table showing correlation between input weights and the proportion of matching recoveries that are successful in the historical data

Score Type	Correlation Coefficient	P-Value
Exact Path Weight	−0.4591	0
Partial Path Weight	0.8900	0
Attribute Weight	−0.6412	0

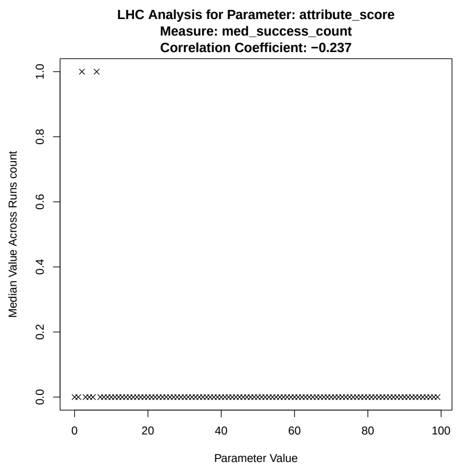

Figure 5.3: Scatter plot showing the relationship between the attribute weight and the number of matching recoveries in the historical data

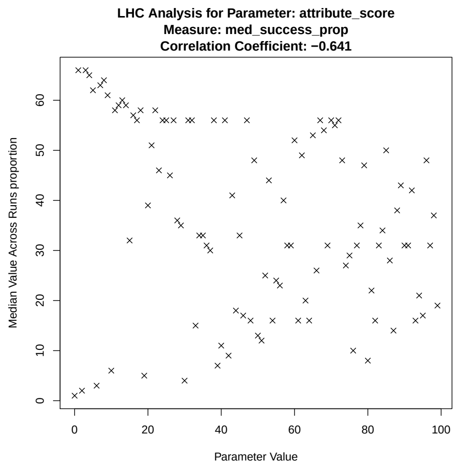

Figure 5.4: Scatter plot showing the relationship between the attribute weight and the number of matching successful recoveries in the historical data

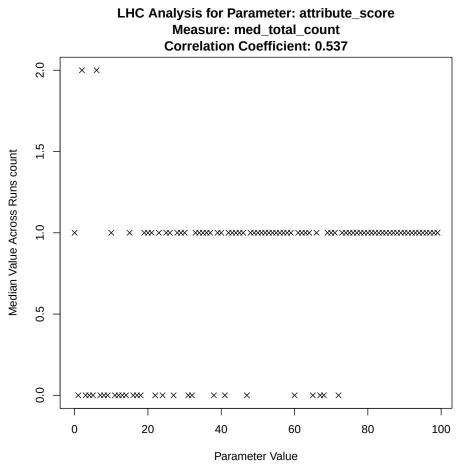

Figure 5.5: Scatter plot showing the relationship between the attribute weight and the proportion of matching recoveries that are successful in the historical data

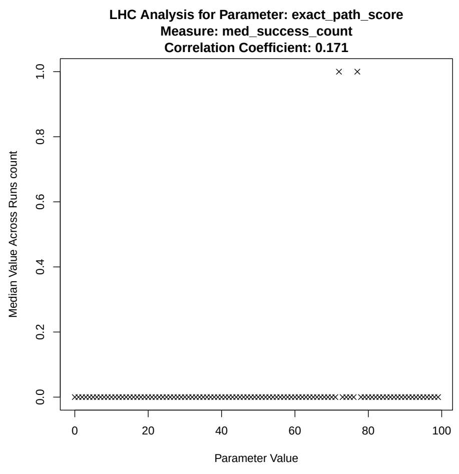

Figure 5.6: Scatter plot showing the relationship between the exact path weight and the number of matching recoveries in the historical data

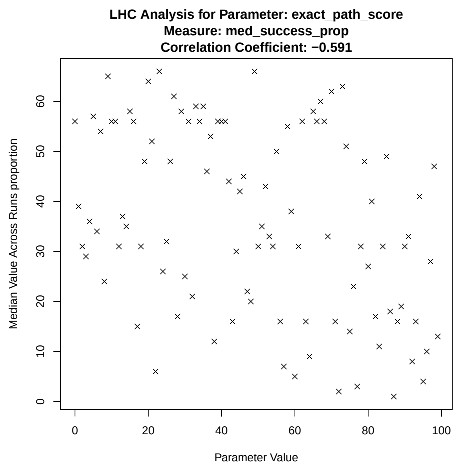

Figure 5.7: Scatter plot showing the relationship between the exact path weight and the number of matching successful recoveries in the historical data

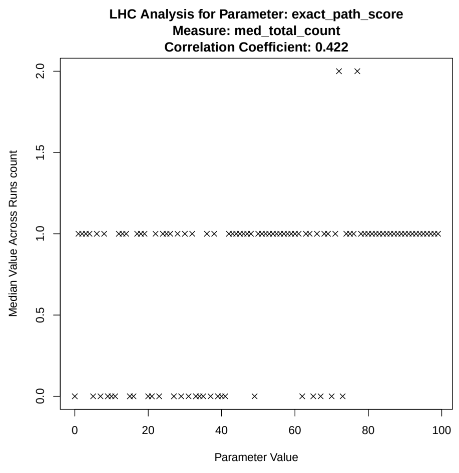

Figure 5.8: Scatter plot showing the relationship between the exact path weight and the proportion of matching recoveries that are successful in the historical data

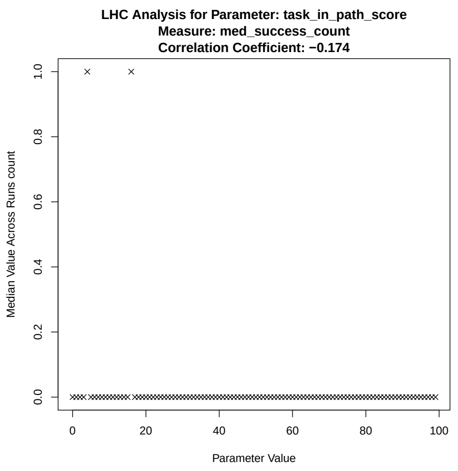

Figure 5.9: Scatter plot showing the relationship between the partial path weight and the number of matching recoveries in the historical data

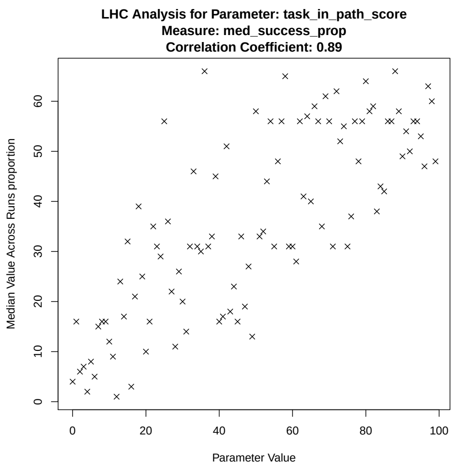

Figure 5.10: Scatter plot showing the relationship between the partial path weight and the number of matching successful recoveries in the historical data

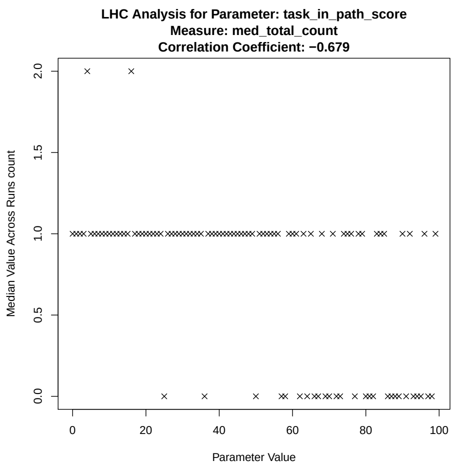

Figure 5.11: Scatter plot showing the relationship between the partial path weight and the proportion of matching recoveries that are successful in the historical data

5.4 Conclusions

This chapter proposes a methodology for extracting recovery strategies from historical process data using a predictive classifier. This is often necessary in business process contexts as the business process design is not often completely specified at a level appropriate to automated reasoning. Some analysis of the generated recoveries is then performed based on the occurrence of the recoveries in historical process data. This leads to the conclusion that the greater the complexity in the business process the proportion of high quality recoveries reduces against the total number of recoveries that can be found.

Once recovery strategies have been discovered the next issue to be solved was that of application; which recovery should be applied to a detected problematic state. A complete autonomic business process system is then proposed using the learning from the prior chapters. This process system can be connected to an existing legacy workflow system and provide automated or semi-automated recovery of predicted business process failures for arbitrary business processes. Weights are used in the process system to tune the selection and application of recovery strategies.

A sensitivity analysis was performed on the weight parameters used for recovery selection in the proposed system. This analysis also used historical data due to the issues with simulating business process systems which have been discussed previously. The conclusion from the sensitivity analysis is that the complexity of the business process is the key determinant of the estimated performance of the proposed process system and that the tested parameter weights do not have a statistically significant effect on the quality of the selected recovery. This is a key finding, as it indicates that procedures which simplify the business process data from the full complexity of the recorded data to a summarised state might yield improved performance in terms of recovery selection. Determining the exact relationship between the complexity of the business process and the performance of the recovery approaches discussed in this chapter would be ideal for future work.

It is likely that businesses would be reluctant to deploy a technology that cannot show a strong expectation of improving the status quo. One of the key issues with the process data used for this analysis is its complexity. The humans involved in the execution of this process are familiar with any inherent complexity and would be better placed to discover solutions to errors than an automated process. If humans could be utilised to produce and rate recoveries, the recovery engine would be able to learn from their knowledge, and use human experience to improve its automated suggestions.

The following chapter evaluates the work in the undertaken in the thesis with discussion of the work conducted and an analysis of the appropriateness of the approaches for business purposes.

Chapter 6

Evaluation & Future Work

This chapter will summarise and evaluate the work undertaken in this thesis from both a scientific and business standpoint. Final conclusions from the work will be discussed in Chapter 7.

Chapter 1 introduces the thesis and the problem space that is to be explored. It sets the contextual background for the rest of the work and provides business context for that work.

Chapter 2 contains a comprehensive review of literature on both business processes an on self-* and autonomic computing. From the review it can be seen that there is little or no prior history of work in the area covered by the research hypothesis. Process improvement techniques are typically taken from successful practitioners (e.g. the Toyota Production System [8] and its derivatives) and converted to generic approaches, or are statistically based, like Six Sigma [7]. Self-* and autonomic approaches in this area are typically based on the infrastructure underlying the business process rather than the process itself, for example, starting additional web servers to satisfy additional demand. There is clearly a gap for a focused technique which attempts to resolve issues with individual process executions.

In order to be able to provide some autonomic response to individual business process executions a predictive technique would be required that would determine which process executions would require an intervention. To that end Chapter 3 examines the performance of well known and well understood classification algorithms from machine learning to determine their performance on business process execution data. Chapter 3 discussed how process data could be gathered and prepared for classification before testing the approaches on real process data from the industrial partner. The chapter concludes that decision trees can provide acceptable performance however they require tuning to do so.

An alternative predictive technique was then sought in order to reduce or remove the requirement for parameter and cost tuning that is needed by the classical classification techniques. Chapter 4 examines immune-inspired techniques which were chosen as they seemed to have characteristics that were well suited to business process data; generally that they are structured to be able to detect many different diverse failure scenarios in a single classifier rather than simply partition in input space. Four immune algorithms were tested against the same process data as used in Chapter 3 and the results discussed. Of the immune classifiers tested Immunos 99 showed the best performance.

Once the process outcome can be predicted, the next step is to determine how the problem can be resolved. Chapter 5 builds upon the classification techniques that are discussed in Chapters 3 and 4 by proposing a technique to link the prediction of a business process execution failure with a recovery strategy that can be used to prevent the failure from occurring. The first problem addressed in Chapter 5 is that of determining recoveries. Business processes are seldom specified completely which makes automated reasoning about the process design virtually impossible. An alternative approach is proposed in which recovery strategies are extracted from historical process data using the classification techniques previously discussed. The second half of Chapter 5 proposes an autonomic process system which utilises the discovered recoveries and a predictive classifier to build a system which can be retrofitted to existing workflow systems to provide self-healing properties to the business processes executed by that workflow system. Both the generation of recovery strategies and the technique used to determine the most appropriate recovery for a given failure scenario are discussed and tested using historical process data.

To evaluate this work it is necessary to consider two different view points. The first is an academic view considering the technology used and the results obtained in isolation. The second is one of business context which is driven by the applicability of the solution to the business providing the data for study. These two view points have different notions of acceptable performance, and of research impact. The scientific evaluation will be undertaken first in the following section and the business evaluation in the section following that.

6.1 Scientific Evaluation

The focus of the thesis has tended towards the self-healing element of the hypothesis and different chapters have examined different aspects of self-healing as it relates to business processes. Each chapter makes contributions to the overall thesis and has its own conclusions which can be drawn. This section will summarise the scientific conclusions from each chapter of the thesis.

Key concerns for the scientific evaluation of this work are the performance of the classification techniques and the performance of the selected recoveries as these are the main indicators of algorithmic suitability. The scientific evaluation is concerned primarily with the optimal performance of the algorithms under examination.

In Chapter 3, investigations into the applicability of classical machine learning classifiers to the problem of business process prediction were performed. Specifically, decision tree and association rule based classifiers were tested. Association rule classifiers were immediately discounted from contention because they showed unacceptable accuracy, which is likely to be due to the noisy nature of the process data. The decision tree classifier had difficulties with the initial data due to the skewed nature of the data set. Using the MetaCost technique to address the imbalance between classification classes gave results that showed for the test data available an accurate classifier can be produced. A weakness of this approach was that the exact cost parameters for MetaCost needed to be determined by trial-and-error which increases the amount of human effort required to utilise the technique on a new business process.

In Chapter 4 a number of immune-inspired classification techniques were evaluated to determine if they could be used as an alternative to the classical techniques. Immune techniques were investigated to determine if they would be able to provide acceptable classification performance without extensive parameter tuning. Of the 4 classifiers that were tested Immunos 99 showed the most favourable performance. Immunos 99 was able to deal more effectively with the process data than the unmodified decision tree algorithms, however it did not perform better than the decisions trees augmented with the MetaCost technique. Of particular concern was the elevated false positive rate exhibited by Immunos 99 compared with the decision tree techniques. An excellent piece of further work would be to examine ways to reduce that false positive rate.

Chapter 5 discusses the discovery and selection of recovery strategies for the autonomic process system. In addition, Chapter 5 also describes a proposed process system that could be used to bring self-healing to existing legacy workflow systems. First, an approach to determining recovery strategies was proposed and tested using historical data and the Immunos 99 classifier from Chapter 4. Second, a mechanism for selecting the most appropriate recovery strategy for a given predicted failure was discussed and a sensitivity analysis of the technique to the process similarity metrics used was performed using historical data to estimate performance. The approaches proposed showed poor performance in terms of the success rate of the chosen recoveries on raw process data (~11% success). The performance of the approach on simplified data was excellent, with the suggested recoveries showing ~84% success. This suggests that simpler processes are more amenable to automated intervention.

A key theme throughout all of the chapters in the thesis was the difficulties that arose around the complexity and chaotic nature of the process data. The simplified data set, which was an extract of the most commonly occurring behaviours, showed better performance throughout. Determining the relationship between process complexity and the performance of the autonomic approach discussed in this thesis is a key question for future work.

In terms of the research hypothesis, is has been shown that self-healing can be applied to business processes using the techniques discussed. However, it has also been shown that the success of these techniques depends mostly upon the complexity of the process being managed.

6.2 Business Evaluation

The most important factor in the business evaluation is the impact of the solution upon business metrics such as overall costs and customer experience. In particular the performance of the classifier will have a large impact on the overall suitability of the system. In this section, the business impact of the results of each chapter of the thesis are considered and then a cost-benefit analysis of the performance of the techniques is used to determine if the techniques are suitable for implementation as they currently stand.

The key concerns for the business evaluation differ slightly from those of the scientific evaluation, which was mainly concerned with algorithmic performance. The business evaluation is concerned with determining the suitability of the techniques in terms of how they could be used in the real world. Businesses are not concerned with the optimal performance of the algorithms, instead they are interested in the cost savings and customer satisfaction increases that these techniques can bring. For example, a classification algorithm that had 0% false positives, and 40% true positives would be considered poor from a scientific standpoint, however this would allow 40% of customers to be better served which would be very acceptable. Generally, the trade off between false positives and true positives is crucial as this is directly related to increasing costs (false positives) and cost savings (true positives). This is illustrated further in the cost/benefit analysis in Section 6.2.1.

In Chapter 3 classical classification techniques are evaluated for their ability to predict the outcome of a business process execution based on execution trace data. Association rules showed unacceptable performance. Decision tree algorithms showed excellent performance where coupled with a technique to deal with the unbalanced nature of the process data (MetaCost in this case). The performance levels would be acceptable to a business in most cases, subject to specific business goals and strategy, however the need for a human to tune the MetaCost parameters for the classifier would limit the applicability of the technique to those critical business processes which could afford to spend resources on parameter tuning.

In Chapter 4 a number of immune-inspired classification techniques were tested to determine if they were able to provide acceptable performance without the requirement for human parameter tuning on new process data. Removing the requirement for tuning would increase the applicability of the technique to a wider number of business processes as the initial setup costs and effort would be lower. One of the four techniques tested showed reasonable performance characteristics - Immunos 99. The true positive performance of Immunos 99 was excellent with a true-positive rate between 68% and 74%, which would provide the opportunity for a significant proportion of failures to be avoided. The false positive rate of the algorithm was a concern though, at between 14.3% and 18.2%. This false positive rate in a process with 10,000 process executions per week would lead to between 1,430 and 1,820 additional investigations. That volume of investigations would be not be able to be absorbed using the existing capacity of the business. Therefore, in high volume processes the number of false positives is likely to make the current performance of these techniques untenable, however these are exactly the processes which would be able to afford tuning of the decision tree/MetaCost solution.

Chapter 5 covers the generation and application of recovery strategies to process failures in a proposed self-healing process system. The approach was tested using both the simplified process data from Chapters 3 and 4 as well as the full data set which the simplified data was synthesised from. The classifier used for the generation of the recoveries was the Immunos 99 classifier from Chapter 4. The results obtained from the testing of the proposed approaches on historical data clearly show there are issues to be addressed. The simplified process data performs well whilst the full data exhibited poor performance. Recoveries on the full data only had a success rate of 11% which would not be sufficient to address the number of interventions which would result from the process predictions. For example, a process with a process with 10,000 process steps per week 1,000 of which would fail without intervention, a classifier with a true positive rate of 74% and a false positive rate of 15.5% would require 2,135 interventions in total. If only 235 of these interventions are successful (assuming 11% success rate) this leaves 1,900 steps pending failure which is much worse than the 1,000 which would fail without intervention. Exactly which success rate would be required for an implementation to be feasible would be dependent on a cost-benefit analysis such as the one discussed in the following section. The results suggest that process complexity is a key determinant for recovery performance. Businesses can typically only move to reduce their inherent complexity of the long-term therefore further research would be required to address the performance drop with increasing complexity before these techniques could be used in at the industrial partner.

6.2.1 Cost-Benefit Analysis

To determine if the techniques described within this thesis would be suitable for development and deployment within a business a cost-benefit analysis would typically be performed. In this section a simulation is used to compare the performance of a standard process system with an autonomic self-healing process system as proposed. The performance results from the earlier chapters are used as parameters to the simulation.

There are a number of costs that are applicable to executing business processes. Those considered for this evaluation are:

- Cost of customer interactions (e.g. incoming calls, letters, email).
- Cost of an automated intervention
- Cost of a manual intervention

Under the assumption that the cost of the tasks are identical between the manual and autonomic cases, the costs incurred by the process can be modelled by considering the number of customer interactions that are involved in a process. For example, if the customer calls to chase the progress of an order, an additional interaction is required. If the customer's order has an intervention the cost of the second interaction is not incurred instead the lower cost of intervention is incurred.

To measure the potential cost savings available through the application of the autonomic system described in this thesis, a simple simulation of a system has been built using the performance profile of the predictive classifier and the estimated success rate of the automated interventions. This simulation covers the intake of jobs over 10 time periods with a nominal 10% failure rate under normal conditions (taken from knowledge of the industrial partner).

In the non-autonomic case, in each time period 90% of the input jobs are successful and 10% carry over into the next time period where they will add to the normal intake of jobs. The number of interactions in each time period can then be calculated using a simple multiplier of £25. Summing the cost of interactions over all 10 periods gives a total cost of the simulation.

For the autonomic case, using the true positive rate and false positive rate of the predictive classifier, the number of jobs that would be identified as requiring intervention can be calculated. Each of these interventions is assigned a cost of £5 to indicate the significantly lower effort that would be required for an internal intervention verses a customer interaction. Using the success percentage from the recovery experimentation allows the number of jobs that would be successfully corrected to be determined. The

Table 6.1: Simulation results showing potential cost savings of using the autonomic process system vs. the standard situation. 10 Task and Full indicate which of the median recovery success rates from Table 5.6 was used in the simulation runs.

Data Set	Min	Lower Quartile	Median	Upper Quartile	Max
J48/10 Task	-948,060	902,705	1,272,130	1,662,023	2,807,820
I99/10 Task	-1,522,955	110,215	505,405	875,876	2,197,010
J48/Full	-2,163,400	-788,998	-381,010	23,611	1,565,160
I99/Full	-2,748,960	-1,352,616	-941,915	-574,685	1,050,510

number of jobs from period P_n that would carry over to period $P_n + 1$ can be calculated as the number of those jobs that fail and are not sent to intervention and the number of jobs that have an unsuccessful intervention. The costs of the interactions and the interventions over the 10 time periods the be calculated to give the total cost of the simulation.

Once the costs of the existing system and the proposed autonomic system are captured from the simulation the delta between the costs can be calculated. This then indicates the change in service costs that result from the autonomic system with the simulated performance characteristics. For the simulations, the number of input jobs in each period is randomly chosen from a normal distribution with the mean 100,000 and the standard deviation of 5000.

Each simulation was executed 1000 times. The results of the simulation runs are shown in Table 6.1 and Figure 6.1. An example of a single run of the simulation to illustrate its construction is shown in Appendix C. In the results presentation I99 stands for the Immunos 99 classifier with a 2-step window, J48 stands for the J48 classifier with a 1-step window. The respective window sizes where chosen because those returned the best performance of the tests performed in Chapter 4 and 3 respectively. The 10 Task or Full label shows which of the recovery result percentages from Chapter 5 was used, 83.77% from the 10 Task simplified process data used in testing the classifiers in Chapters 3 and 4 or 11.11% from the full process data set with no simplifications (see Table 5.6).

The results of the simulation show that under the assumptions of the simulation, there are significant cost savings to be made for the simpler 10 Task data sets. However, the current situation is closest to the two Full scenarios, where the classification performance and recovery success performance are both low. Further work to investigate how data and process complexity correlates with process performance, and to determine how complex process data is best processed for consumption by classification algorithms would be required to improve performance before any implementation would be appropriate

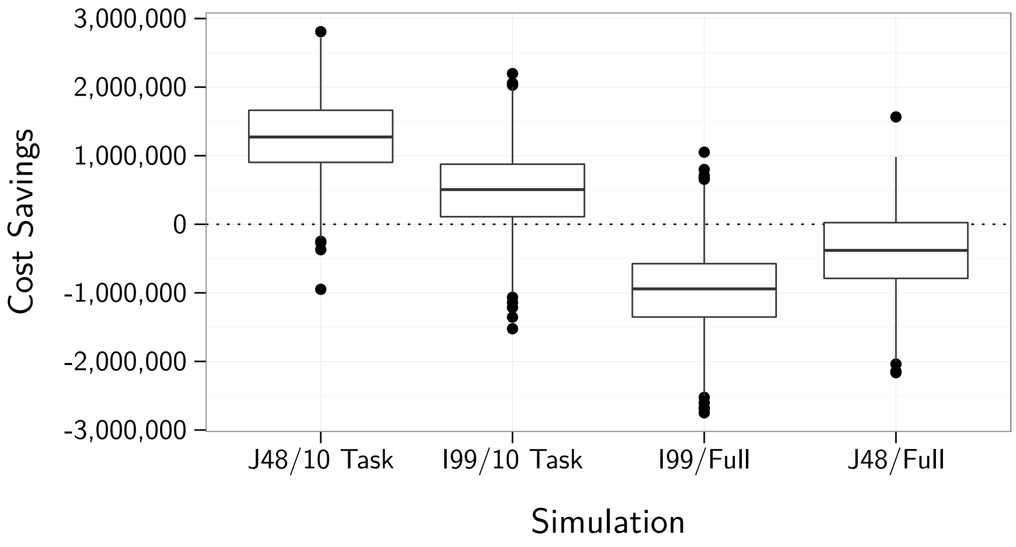

Figure 6.1: Box plot showing distributions of cost savings in simulation scenarios. 10 Task and Full indicate which of the median recovery success rates from Table 5.6 was used in the simulation runs.

6.3 Future Direction

Each chapter of the thesis has briefly discussed possible future research directions for the work covered within. The key theme for future work is dealing with the complexity of realistic business process data as this as a detrimental impact on the classification and recovery performance throughout. The next steps at BT will be to attempt to understand and address these questions in order to build a system suitable for use. The approach will need to be tested and validated within a limited trial to confirm its suitability for use in a wider context.

First a candidate business process will be identified for a trial of the technology. The first step will then be an analysis of the performance of the classification algorithms for that process. This analysis will include lengthy consideration on the characterisation of the training data for the classifier and on mechanisms for dealing with the inherent complexity of the data. For example, it may benefit the classifiers performance to only train upon the most popular process behaviour.

The next step will be to implement the classifier on the existing live business process so that its performance can be monitored over a considerable period of time. During the test no automatic intervention will be performed, instead lists of potential failures will be provided to the business for manual investigation. Feedback on the quality of that list will be essential in tuning the classifier for real use. The system will be providing real business benefit at this point by allowing the company to resolve process issues before they impact customers, raising customer satisfaction and lowering the average cost to serve for the monitored process.

Once the classifier is suitable for the identification of process issues the next step will be to produce recovery strategies using the classifier and the technique discussed in Chapter 5. It will then be possible to make a recovery suggestion for each of the identified process issues. The failure list will then be extended to include a suggestion for how the business process might be corrected in each case. Feedback from the agents investigating the potential failures will be used to guide development and modification of the recovery generation and selection algorithms.

Finally, once the feedback and metrics from the agents using the feedback suggestions reaches a suitably high level, work can begin to integrate the recovery paths automatically into the workflow for impacted business processes. A confidence metric will need to be developed to allow the system to determine for each case whether it should be automatically recovered, or if it should be sent to a human for manual investigation. Over time as the technique improves the number of manual interventions should fall.

The integration of the system proposed in this thesis with a running workflow system will require a lengthy and costly change programme and therefore an extremely high level of confidence in the benefits of the autonomic process system. Taking a step by step approach as outlined here will allow the system to be rigorously tested at each stage.

6.4 Summary

This thesis has examined in detail the issues that arise in realising an autonomic business process system that is able to monitor itself for potential failures and then to self-correct those potential failures by changing their execution paths. The research has concluded that it is possible to predict imminent business process failures using both classical classification techniques and immune-inspired techniques and discovered the weaknesses of each of those approaches. Further, the work has concluded that it is possible to discover and assign recovery strategies to impending failures in the absence of a model or design of the process being monitored. However the investigations have determined that the success of the automated recoveries depends largely upon the complexity of the process under control.

The thesis has also considered the commercial applicability of the outcomes of the research though a simulation and associated cost-benefit analysis. The cost-benefit analysis shows that an autonomic process system can reduce both an organisations cost to serve and the number of failures which make it though to a customer impacting event. Finally, the thesis has shown that such benefit is more likely to be realised on less complex business processes.

Chapter 7

Conclusions

7.1 Research Hypotheses

The work discussed in this thesis was intended to address the research questions raised by the hypothesis and sub-hypotheses introduced in Section 1.4. These hypotheses are as follows:

> Business process improvement can incorporate autonomic properties of self-healing to provide solutions to changing business contexts.

This hypothesis was decomposed into 3 sub-hypotheses as follows:

Hypothesis 1: Using historical business process data it is possible to predict the outcome of a business process using a classical classification technique (Chapter 3).

Hypothesis 2: It is possible to produce a better performing classifier to predict the outcome of a business process using immune-inspired classification techniques rather than classical classification techniques (Chapter 4).

Hypothesis 3: It is possible to produce accurate and useful recovery plans for business processes in the absence of a formal model of the business process (Chapter 5).

Each of the technical chapters addresses one of the aforementioned sub-hypotheses. This chapter will bring together the contributions of each chapter to the resolution of the hypotheses and the contribution to knowledge and contribution to the industrial partner.

7.2 Process Prediction using Classical Techniques

Chapter 3 contributes a method to exploit historical business process data to produce a classifier suitable for the prediction of the outcome for executing business processes. This method was tested against simplified data from the industrial partner and found to provide acceptable performance when paired with a technique to address class skew in the training data set.

The key contribution of this chapter is to inform businesses that relatively well known techniques can be used to provide outcome prediction to generic business processes using data that would be captured as part of the existing workflow systems. This tells the business that such techniques can be implemented with a few changes required to existing systems.

With respect to the first sub-hypothesis, work in Chapter 3 shows that it is possible to predict the outcome of a business process execution using classical techniques.

7.3 Process Prediction using Immune Inspired Techniques

Chapter 4 builds upon the technique used to produce classical classifiers in Chapter 3 to apply the process data to a selection of immune-inspired classification algorithms. This chapter provides a number of contributions to knowledge in addition to the contribution to the industrial partner.

The first contribution to knowledge is the application of immune-inspired algorithms to business process prediction. This is a new area of application for these techniques that has not been previously explored.

The second contribution is an important contribution to the industrial partner. Chapter 4 tests a variety of immune-inspired classification techniques and determines that Immunos 99 provides the best performance albeit with an elevated false positive rate. In addition to informing the industrial partner about which techniques are best suited for process prediction this also exposes BT to a number of new classification techniques that could be used in a wider set of application domains within the business.

An additional contribution to business of this chapter is the in depth case study of applying an immune inspired classification techniques to a specific a BT process. The results of the study were communicated back to interested parties inside the company. This aids BT in making decisions on its own research programme. As a result of this case study interest in immune-inspired techniques has grown and will certainly form part of future work in a variety of problem domains such as churn prediction.

With respect to the second sub-hypothesis Chapter 4 shows that it is possible to produce a classifier for business process prediction using immune-inspired techniques but fails to prove that these are definitively better than the classical classifiers, primarily due to the elevated false positive rate for the immune classifiers.

7.4 Determining Process Recovery Paths

Chapter 5 addresses the third of the sub-hypotheses, that of producing an appropriate recovery plan for a failing business process in the absence of a defined model of the business process. Results of applying this technique to example data from the industrial partner show that it performs better on simpler business processes and performance is poorer on more complex data. The proposed method is applicable to all business processes.

A measure of the impact of this work at the industrial partner is that significant effort was spent determining the patentability of this invention. Whilst the determination was ultimately that this invention does not qualify for patent protection work will continue within the research programme aiming to produce a prototype that can demonstrate the technology for internal and external use.

The proposed recovery method is also a contribution to knowledge given the absence of other techniques able to deal with systems without a formal model or a manually produced list of recoveries. Whilst the technique has been discussed with reference to business processes, it may be possible to apply the same or a similar approach to diverse problem spaces where no formal model of the system exists.

7.5 Business Contribution vs. Scientific Contribution

Given the industrial nature of the EngD programme this thesis has needed to consider the contributions it makes to the industrial partner as well as those contributions it makes to knowledge. In places there is a trade off between these two facets. As shown in the other sections of this chapter there is generally a strong synergy between the contributions to knowledge and contribution to business, both communities are informed by the work in the thesis.

Some sections of this thesis report primarily on business concerns. For example, the cost-benefit model for the proposed autonomic process system discussed in Section

6.2.1 is much more important in a business context than an academic one. Whilst the contribution to knowledge made by this thesis is important, it was the business context that guided the development, scope and scale of the work undertaken.

Appendices

Appendix A

High Volume Wholesale Service Provision Classification Results

These are the raw results reported by Weka for the 10-fold cross-validation performed on the data set discussed in the case study in Section 4.6

Summary

Correctly Classified Instances	75433	75.433 %
Incorrectly Classified Instances	24567	24.567 %
Kappa statistic	0.2122	
Mean absolute error	0.2457	
Root mean squared error	0.4957	
Relative absolute error	129.1558 %	
Root relative squared error	160.7236 %	
Coverage of cases (0.95 level)	75.433 %	
Mean rel. region size (0.95 level)	50 %	
Total Number of Instances	100000	

Confusion Matrix

a	b	<– classified as
6086	4557	a = outcome=N
20010	69347	b = outcome=Y

Appendix B

Spartan Analysis Candidate Scores

	Exact Path Match Score	Task in Path Match Score	Attribute Match Score
1	21	80	9
2	34	62	25
3	75	31	88
4	99	96	80
5	77	16	7
6	92	51	46
7	51	60	70
8	96	21	76
9	48	28	82
10	16	89	22
11	46	39	27
12	54	38	35
13	94	64	43
14	85	71	91
15	35	82	14
16	44	9	65
17	4	25	75
18	36	33	24
19	94	46	61
20	66	81	19
21	19	79	74
22	10	59	4
23	3	24	60
24	73	5	3
25	97	62	86

	Exact Path Match Score	Task in Path Match Score	Attribute Match Score
26	88	13	0
27	58	74	72
28	25	15	16
29	70	72	5
30	81	49	75
31	19	38	92
32	11	95	42
33	54	32	36
34	84	28	40
35	77	44	57
36	57	4	39
37	13	76	99
38	10	93	68
39	50	89	3
40	86	43	44
41	33	66	13
42	72	2	82
43	16	25	18
44	70	46	36
45	74	98	8
46	5	26	29
47	17	8	33
48	96	0	31
49	41	70	32
50	6	65	16
51	61	7	19
52	22	74	61
53	14	22	30
54	25	29	66
55	31	68	48
56	66	80	71
57	2	18	21
58	29	51	12
59	29	15	47
60	43	54	54
61	85	90	62
62	23	2	10
63	90	47	100
64	26	57	49

	Exact Path Match Score	Task in Path Match Score	Attribute Match Score
65	9	13	55
66	46	85	92
67	78	56	77
68	51	68	79
69	79	99	96
70	41	87	38
71	88	41	48
72	0	77	72
73	33	18	94
74	38	11	51
75	57	21	55
76	65	11	42
77	62	87	32
78	24	36	1
79	63	9	93
80	74	42	21
81	27	69	10
82	55	92	85
83	61	61	84
84	48	30	63
85	83	41	95
86	8	91	68
87	67	99	14
88	69	54	26
89	31	20	52
90	92	6	81
91	37	96	65
92	60	83	89
93	39	58	27
94	82	65	57
95	91	75	98
96	45	36	38
97	7	53	84
98	13	34	59
99	99	50	50
100	52	85	89

Appendix C

Cost Saving Simulation Example

Is shown overleaf.

Parameters:		StdDev	
Input Job Count	100,000	5000	We are assuming there is enough capacity for these
Failure %	10%		
Cost per Interaction	£ 25.00		
Autonomic TP %	87%		For Failures - What % of failures we would catch correctly
Autonomic FP %	3%		For Failures - What % of succcesses we would incorrectly class as failures
Autonomic Intervention Accuracy	11%		What percentage of the automated interventions would actually succeed and not repeat
Cost per Intervention	£ 5.00		

Cost Reduction	-£ 908,300
Customer Driven Interaction Reduction	- 11,646
Failures Prevented	9,460

Non Autonomic Case											Totals
Iteration	1	2	3	4	5	6	7	8	9	10	
Jobs repeating from Prev Stage	-	9,157	11,447	11,511	11,148	11,158	11,133	10,371	11,125	11,295	98,345
New Jobs (Identical in both cases)	91,578	103,195	103,703	112,639	105,571	104,661	98,822	108,106	98,114	87,490	
Total Jobs Presented	91,578	114,477	115,113	111,487	111,589	111,330	103,713	111,258	112,950	109,807	1,093,302
Jobs Succeeded	82,421	103,030	103,602	100,339	100,431	100,197	93,342	100,133	101,655	98,827	983,977
Jobs that will Repeat	9,157	11,447	11,511	11,148	11,158	11,133	10,371	11,125	11,295	10,980	109,325
Cost of Interactions	£ 2,289,450	£ 2,861,925	£ 2,877,825	£ 2,787,175	£ 2,789,725	£ 2,783,250	£ 2,592,825	£ 2,781,450	£ 2,823,750	£ 2,745,175	£ 27,332,550

Autonomic Case											
Iteration	1	2	3	4	5	6	7	8	9	10	
Jobs Repeated from Prev Stage	-	8,276	10,075	10,283	11,110	10,546	10,412	9,873	10,663	9,831	91,069
New Jobs (Identical in both cases)	91,578	103,195	103,703	112,639	105,571	104,661	98,822	108,106	98,114	87,490	
Total Jobs Presented	91,578	111,471	113,778	122,922	116,681	115,207	109,234	117,979	108,777	97,321	1,104,948
Jobs that would have Succeded Normally	82,421	100,324	102,401	110,630	105,013	103,687	98,311	106,182	97,900	87,589	994,458
Jobs incorrectly sent to intervention	2,308	2,810	2,868	3,098	2,941	2,904	2,753	2,974	2,742	2,453	27,851
Jobs correctly sent to intervention	7,921	9,643	9,842	10,633	10,093	9,965	9,449	10,205	9,409	8,419	95,579
Total Interventions	10,229	12,453	12,710	13,731	13,034	12,869	12,202	13,179	12,151	10,872	123,430
Successful & Correct Interventions	881	1,072	1,094	1,182	1,122	1,108	1,050	1,134	1,046	936	
Jobs that will Repeat	8,276	10,075	10,283	11,110	10,546	10,412	9,873	10,663	9,831	8,796	99,865
Cost of Original Interactions	£ 2,289,450	£ 2,786,775	£ 2,844,450	£ 3,073,050	£ 2,917,025	£ 2,880,175	£ 2,730,850	£ 2,949,475	£ 2,719,425	£ 2,433,025	£ 27,623,700
Cost of Interventions	£ 51,145	£ 62,265	£ 63,550	£ 68,655	£ 65,170	£ 64,345	£ 61,010	£ 65,895	£ 60,755	£ 54,360	£ 617,150
Total	£ 2,340,595	£ 2,849,040	£ 2,908,000	£ 3,141,705	£ 2,982,195	£ 2,944,520	£ 2,791,860	£ 3,015,370	£ 2,780,180	£ 2,487,385	£ 28,240,850

Bibliography

[1] P. Taylor, M. Leida and B. A. Majeed, ‘Case Study in Process Mining in a Multinational Enterprise’, in *Data-Driven Process Discovery and Analysis*, ser. Lecture Notes in Business Information Processing, K. Aberer, E. Damiani and T. Dillon, Eds., vol. 116, Springer Berlin Heidelberg, 2012, pp. 134–153.

[2] M. Spott, D. Nauck and P. Taylor, ‘Modern Analytics in Field and Service Operations’, in *Transforming Field and Service Operations*, G. Owusu, P. O’Brien, J. McCall and N. F. Doherty, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, ch. Part II, pp. 85–99.

[3] P. Taylor, F. Polack and J. Timmis, ‘Accelerating Immunos 99’, in *Advances in Artificial Life, ECAL*, vol. 12, MIT Press, Sep. 2013, pp. 893–898.

[4] P. Taylor, ‘Autonomic Process Management for BT’, BT Innovate and Design, Internal Report, Mar. 2012.

[5] P. Taylor, ‘Process Prediction for Openreach’, BT Research and Innovation, Internal Report, 2014.

[6] P. Taylor, ‘Autonomic process business impact model’, BT Technology, Service and Operations, Internal Report, 2014.

[7] T. Pyzdek, *The Six Sigma Handbook*. McGraw-Hill, 2003.

[8] T. Ohno, *Toyota Production System: Beyond Large-scale Production*. Productivity Press, 1988, p. 152.

[9] W. A. Shewhart, *Economic Control of Quality of Manufactured Product*, 7. D. Van Nostrand Company, Inc, 1931, vol. 30, p. 501.

[10] Red Hat Inc., *JBoss jBPM*. `http://www.jboss.org/jbpm`.

[11] Progress Software Corporation, *Progess Savvion*. `http://web.progress.com/en/savvion/`.

[12] Oracle Corporation, *Oracle Business Intelligence*. `http://www.oracle.com/us/solutions/ent-performance-bi/index.html`.

[13] Lightfoot Solutions, *Signals from Noise (SfN)*. `http://www.lightfootsolutions.com/signalsfromnoise_sfn.html`.

[14] W. M. P. van der Aalst, *Process Mining*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011.

[15] B. Miller, *Can you CHOP up autonomic computing?*, 2005. `http://www.ibm.com/developerworks/library/ac-edge4/index.html` (visited on 01/08/2011).

[16] B. Jacob, *A practical guide to the IBM autonomic computing toolkit*. IBM, International Technical Support Organization, 2004.

[17] A. Smith, *An Inquiry into the Nature and Causes of the Wealth of Nations*, R. H. Campbell and A. S. Skinner, Eds. Liberty Fund, 1776, vol. 2, ch. 10, pp. 1–593.

[18] M. Hammer and J. Champy, *Reengineering the Corporation: A Manifesto for Business Revolution*, ser. Harper Business 5. HarperCollins, 1993, vol. 36, pp. 90–91.

[19] T. H. Davenport, *Process innovation: reengineering work through information technology*, H. B. S. Press, Ed., 1. Harvard Business School Press, 1993, vol. 11, p. 337.

[20] *Workflow Management Coalition*. `http://www.wfmc.org/`.

[21] Workflow Management Coalition, 'Workflow Management Coalition Terminology & Glossary', no. 3, 1999. `http://www.wfmc.org/Download-document/WFMC-TC-1011-Ver-3-Terminology-and-Glossary-English.html`.

[22] D. Simonovich, *From Inter-organizational to Inter-departmental Collaboration – Using Multiple Process Levels*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2006, vol. 4277, pp. 854–862.

[23] J. O. Kephart and D. M. Chess, 'The vision of autonomic computing', *IEEE Computer*, vol. 36, no. 1, pp. 41–50, Jan. 2003.

[24] T. A. Aldowaisan and L. K. Gaafar, 'Business process reengineering: an approach for process mapping', *Omega*, vol. 27, no. 5, pp. 515–524, Oct. 1999.

[25] C. Perrow, *Normal Accidents*, N. J. Smelser and P. B. Baltes, Eds., 7365. Basic Books, 1984, vol. 477, pp. 33–38.

[26] H. A. Reijers, S. Limam and W. M. P. van der Aalst, 'Product-Based Workflow Design', *Journal of Management Information Systems*, vol. 20, no. 1, pp. 229–262, Jul. 2003.

[27] K. van Hee, L. J. Somers and M. Voorhoeve, 'The ExSpecT tool', in *VDM'91 Formal Software Development Methods*, ser. Lecture Notes in Computer Science, S. Prehn and W. Toetenel, Eds., vol. 551, Springer Berlin / Heidelberg, Oct. 1991, pp. 683–684.

[28] J. Mendling and C. Simon, 'Business Process Design by View Integration', *Lecture Notes in Computer Science*, vol. 4103, pp. 55–64, 2006.

[29] Oracle Corporation, *Oracle Business Process Management*. `http://www.oracle.com/us/technologies/bpm/`.

[30] R. Shah, 'Lean manufacturing: context, practice bundles, and performance', *Journal of Operations Management*, vol. 21, no. 2, pp. 129–149, Mar. 2003.

[31] M. Allway and S. Corbett, 'Shifting to lean service: Stealing a page from manufacturers' playbooks', *Journal of Organizational Excellence*, vol. 21, no. 2, pp. 45–54, 2002.

[32] P. Ahlstrom, 'Lean service operations: translating lean production principles to service operations', *International Journal of Services Technology and Management*, 2004.

[33] Pentaho Corporation, *Pentaho BI Suite*. `http://www.pentaho.com`.

[34] B. Azvine, Z. Cui, D. Nauck and B. A. Majeed, 'Real Time Business Intelligence for the Adaptive Enterprise', in *The 8th IEEE International Conference on E-Commerce Technology and The 3rd IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services (CEC/EEE'06)*, IEEE, Jun. 2006, pp. 29–29.

[35] Mozilla Foundation, *Bugzilla*. `http://www.bugzilla.org`.

[36] W. M. P. van der Aalst, A. H. M. Ter Hofstede and M. Weske, *Business Process Management: A Survey*, 2003.

[37] D. Jordan and J. Evdemon, *Web Services Business Process Execution Language Version 2.0. OASIS Standard*, 2007.

[38] TIBCO Software Inc, 'The Case for Business Process Management', `http://www.tibco.com/resources/software/bpm/case%5C_for%5C_bpm%5C_wp.pdf`.

[39] Software AG, *ARIS Business Architect & Designer*. `http://www.softwareag.com/corporate/products/aris_platform/aris_design/business_architect/overview/`.

[40] Object Management Group, Inc., *Business Process Model And Notation (BPMN) Version 2.0*. `http://www.omg.org/spec/BPMN/2.0/`.

[41] A. J. M. M. Weijters and W. M. P. van der Aalst, 'Process Mining Discovering Workflow Models from Event-Based Data', *Management*, no. i, B. Kröse, M. Rijke, G. Schreiber and M. Someren, Eds., pp. 283–290, 2001.

[42] A. Tiwari, C. Turner and B. A. Majeed, 'A review of business process mining: state-of-the-art and future trends', *Business Process Management Journal*, vol. 14, no. 1, pp. 5–22, Aug. 2008.

[43] Eindhoven Technical University, *The ProM Framework*. `http://www.promtools.org/prom5/`.

[44] W. M. P. van der Aalst, H. A. Reijers and M. Song, 'Discovering Social Networks from Event Logs', *Computer Supported Cooperative Work (CSCW)*, vol. 14, no. 6, pp. 549–593, Oct. 2005.

[45] A. Rozinat and W. M. P. van der Aalst, 'Decision mining in ProM', *Business Process Management*, Lecture Notes in Computer Science, vol. 4102, pp. 420–425, 2006.

[46] K. Batoulis, A. Meyer, E. Bazhenova, G. Decker and M. Weske, 'Extracting decision logic from process models', English, in *Advanced Information Systems Engineering*, ser. Lecture Notes in Computer Science, J. Zdravkovic, M. Kirikova and P. Johannesson, Eds., vol. 9097, Springer International Publishing, 2015, pp. 349–366.

[47] A. Senderovich, M. Weidlich, A. Gal and A. Mandelbaum, 'Mining resource scheduling protocols', English, in *Business Process Management*, ser. Lecture Notes in Computer Science, S. Sadiq, P. Soffer and H. Völzer, Eds., vol. 8659, Springer International Publishing, 2014, pp. 200–216.

[48] R. Sarno, P. Sari, H. Ginardi, D. Sunaryono and I. Mukhlash, 'Decision mining for multi choice workflow patterns', in *2013 International Conference on Computer, Control, Informatics and Its Applications (IC3INA)*, Nov. 2013, pp. 337–342.

[49] Fluxicon, *Disco*. `http://www.fluxicon.com/`.

[50] J. Eisner, *State-of-the-art algorithms for minimum spanning trees: A tutorial discussion*, 1997. `http://www.cs.jhu.edu/~jason/papers/eisner.mst-tutorial.pdf` (visited on 29/10/2011).

[51] E. Kandogan and J. Bailey, 'Usable Autonomic Computing Systems: The Administrator's Perspective', pp. 18–26, May 2004.

[52] G. Di Marzo Serugendo, N. Foukia, S. Hassas, A. Karageorgos, S. Mostéfaoui, O. Rana, M. Ulieru, P. Valckenaers and C. Van Aart, 'Self-Organisation: Paradigms and Applications', in *Engineering Self-Organising Systems*, ser. Lecture Notes in Computer Science, G. Di Marzo Serugendo, A. Karageorgos, O. Rana and F. Zambonelli, Eds., vol. 2977, Springer Berlin / Heidelberg, 2004, pp. 1–19.

[53] M. Wooldridge and N. R. Jennings, 'Intelligent agents: Theory and practice', *Knowledge engineering review*, vol. 10, no. 2, pp. 115–152, 1995.

[54] X. Du, W. Song and M. Munro, 'Semantics Recognition in Service Composition Using Conceptual Graph', in *2006 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology Workshops*, IEEE, Dec. 2006, pp. 295–298.

[55] D. Martin, M. Burstein, J. Hobbs, O. Lassila, D. McDermott, S. McIlraith, S. Narayanan, M. Paolucci, B. Parsia, T. Payne *et al.*, 'OWL-S: Semantic markup for web services', 2004. `http://www.ai.sri.com/daml/services/owl-s/1.2/overview/`.

[56] C. K. Liu and D. Booth, 'Web services description language (WSDL) version 2.0 part 0: Primer', W3C, W3C Recommendation, Jun. 2007. `http://www.w3.org/TR/2007/REC-wsdl20-primer-20070626`.

[57] A. Blum and M. Furst, 'Fast planning through planning graph analysis', *Artificial intelligence*, vol. 90, no. 1-2, pp. 281–300, 1997.

[58] E. L. Ulungu and J. Teghem, 'Multi-objective combinatorial optimization problems: A survey', *Journal of Multi-Criteria Decision Analysis*, vol. 3, no. 2, pp. 83–104, 1994.

[59] E. Bonabeau, M. Dorigo and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*, ser. Santa Fe Institute Studies on the Sciences of Complexity 1. Oxford University Press, USA, 1999, vol. 4, p. 320.

[60] E. Bonabeau, M. Dorigo and G. Theraulaz, *Swarm Intelligence - From Natural to Artificial Systems*, ser. Studies in the sciences of complexity. Oxford University Press, 1999.

[61] A. F. T. Winfield, C. J. Harper and J. Nembrini, 'Towards Dependable Swarms and a New Discipline of Swarm Engineering', in *Lecture Notes in Computer Science*, ser. Lecture Notes in Computer Science, E. Şahin and W. Spears, Eds., vol. 3342, Springer, 2005, pp. 126–142.

[62] E. Şahin, 'Swarm robotics: From sources of inspiration to domains of application', in *Proceedings of the 2004 International Conference on Swarm Robotics*, ser. SAB'04, Santa Monica, CA: Springer-Verlag, 2005, pp. 10–20.

[63] E. Tuci, B. Mitavskiy and G. Francesca, 'On the evolution of self-organised role-allocation and role-switching behaviour in swarm robotics: A case study', in *Proceedings of the Twelfth European Conference on the Synthesis and Simulation of Living Systems: Advances in Artificial Life, ECAL 2013, Sicily, Italy, September 2-6, 2013*, 2013, pp. 379–386.

[64] F. Polack, 'Self-organisation for survival in complex computer architectures', in *Proceedings of the First international conference on Self-organizing architectures*, Springer, 2010, pp. 66–83.

[65] J. Timmis, A. Tyrrell, M. Mokhtar, A. Ismail, N. Owens and R. Bi, 'An artificial immune system for robot organisms', *Symbiotic Multi-Robot Organisms: Reliability, Adaptability, Evolution*, pp. 268–288, 2010.

[66] N. Capodieci, E. Hart and G. Cabri, 'An immune network approach for self-adaptive ensembles of autonomic components: A case study in swarm robotics', in *Proceedings of the Twelfth European Conference on the Synthesis and Simulation of Living Systems: Advances in Artificial Life, ECAL 2013, Sicily, Italy, September 2-6, 2013*, 2013, pp. 864–871.

[67] Y. Khaluf and F. J. Rammig, 'Task allocation strategy for time-constrained tasks in robot swarms', in *Proceedings of the Twelfth European Conference on the Synthesis and Simulation of Living Systems: Advances in Artificial Life, ECAL 2013, Sicily, Italy, September 2-6, 2013*, 2013, pp. 737–744.

[68] R. Kota, N. Gibbins and N. R. Jennings, 'Decentralised Approaches for Self-Adaptation in Agent Organisations', *ACM Transactions on Autonomous and Adaptive Systems*, vol. 7, no. 1, 1:1–1:28, May 2012.

[69] H. Parunak, '"Go to the ant": Engineering principles from natural multi-agent systems', *Annals of Operations Research*, vol. 75, pp. 69–102, 1997.

[70] D. Weyns, K. Schelfthout and T. Holvoet, 'Exploiting a virtual environment in a real-world application', in *Environments for Multi-Agent Systems II, Second International Workshop, E4MAS 2005, Utrecht, The Netherlands, July 25, 2005, Selected Revised and Invited Papers*, 2005, pp. 218–234.

[71] G. Tesauro, D. Chess, W. Walsh, R. Das, A. Segal, I. Whalley, J. Kephart and S. White, 'A multi-agent systems approach to autonomic computing', in *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 1*, IEEE Computer Society, 2004, pp. 464–471.

[72] K. Fischer, 'Self-organisation in Holonic Multiagent Systems', in *Mechanizing Mathematical Reasoning*, ser. Lecture Notes in Computer Science, D. Hutter and W. Stephan, Eds., vol. 2605, Springer Berlin / Heidelberg, 2005, pp. 543–563.

[73] M. Schillo and K. Fischer, 'Holonic Multiagent Systems', *Artificial Intelligence*, vol. 8, no. 13, pp. 1–2, 2002.

[74] T. Itao, T. Suda and T. Aoyama, 'Jack-in-the-net: Adaptive networking architecture for service emergence', in *Proceedings of the Asian-Pacific Conference on Communications*, vol. 9, 2001.

[75] D. Dasgupta, 'An artificial immune system as a multi-agent decision support system', in *Systems, Man, and Cybernetics, 1998. 1998 IEEE International Conference on*, vol. 4, Oct. 1998, 3816–3820 vol.4.

[76] S. Sathyanath and F. Sahin, 'Application of artificial immune system based intelligent multi agent model to a mine detection problem', in *2002 IEEE International Conference on Systems, Man and Cybernetics*, vol. 3, Oct. 2002.

[77] A. Watkins, J. Timmis and L. C. Boggess, 'Artificial Immune Recognition System (AIRS) : An Immune Inspired Supervised Machine Learning Algorithm', *Genetic Programming and Evolvable Machines*, vol. 5, no. 3, pp. 291–317, 2004.

[78] J. Brownlee, 'Immunos-81, The Misunderstood Artificial Immune System', Technical report No. 1-02, Swinburne University of Technology, Austrailia, 2005.

[79] R. Fikes and N. Nilsson, 'STRIPS: A new approach to the application of theorem proving to problem solving', *Artificial intelligence*, vol. 2, no. 3-4, pp. 189–208, 1972.

[80] D. Nau, Y. Cao, A. Lotem and H. Muñoz-Avila, 'SHOP: Simple hierarchical ordered planner', in *Proceedings of the 16th international joint conference on Artificial intelligence-Volume 2*, Morgan Kaufmann Publishers Inc., 1999, pp. 968–973.

[81] S. Thompson, N. Giles, Y. Li, H. Gharib and T. Nguyen, 'Using AI and semantic web technologies to attack process complexity in open systems', *Knowledge-Based Systems*, vol. 20, no. 2, pp. 152–159, 2007.

[82] R. Lundh, L. Karlsson and A. Saffiotti, 'Dynamic self-configuration of an ecology of robots', in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, Oct. 2007, pp. 3403–3409.

[83] A. Aamodt and E. Plaza, 'Case-based reasoning: Foundational issues, methodological variations, and system approaches', *AI communications*, vol. 7, no. 1, pp. 39–59, 1994.

[84] M. M. J. Khan, M. M. Awais and S. Shamail, 'Enabling Self-Configuration in Autonomic Systems Using Case-Based Reasoning with Improved Efficiency', in *Fourth International Conference on Autonomic and Autonomous Systems (ICAS'08)*, IEEE, Mar. 2008, pp. 112–117.

[85] M. Fox and D. Long, 'Fast temporal planning in a Graphplan framework', *Proceedings of AIPS Workshop on Planning for Temporal Domains*, pp. 9–17, 2002.

[86] P. Traverso and M. Pistore, 'Automated composition of semantic web services into executable processes', *The Semantic Web–ISWC 2004*, pp. 380–394, 2004.

[87] F. Giunchiglia and P. Traverso, 'Planning as model checking', *Recent Advances in AI Planning*, 2000.

[88] Y. Yu, H. Ma and M. Zhang, 'An adaptive genetic programming approach to qos-aware web services composition', in *Proceedings of the IEEE Congress on Evolutionary Computation, CEC 2013, Cancun, Mexico, June 20-23, 2013*, 2013, pp. 1740–1747.

[89] J. C. Knight and E. A. Strunk, 'Achieving critical system survivability through software architectures', *Architecting Dependable Systems II*, pp. 69–91, 2004.

[90] M. C. Huebscher and J. A. McCann, 'A survey of autonomic computing - degrees, models, and applications', *ACM Comput. Surv.*, vol. 40, no. 3, 2008.

[91] D. Garlan, S. Cheng and B. Schmerl, 'Increasing system dependability through architecture-based self-repair', *Architecting Dependable Systems*, pp. 61–89, 2003.

[92] J. Park, G. Yoo, C. Jeong and E. Lee, 'Self-management System Based on Self-healing Mechanism', in *Management of Convergence Networks and Services*, ser. Lecture Notes in Computer Science, Y.-T. Kim and M. Takano, Eds., vol. 4238, Springer Berlin / Heidelberg, 2006, pp. 372–382.

[93] D. Garlan and B. Schmerl, 'Model-based adaptation for self-healing systems', in *Proceedings of the first workshop on Self-healing systems - WOSS '02*, New York, New York, USA: ACM Press, Nov. 2002, p. 27.

[94] S. Stepney, R. Smith, J. Timmis and A. Tyrrell, 'Towards a conceptual framework for artificial immune systems', *Artificial Immune Systems*, 2004.

[95] S. H. Russ, A. Lambert, R. King, R. Rajan and D. Reese, 'An artificial immune system model for task allocation', in *Proceedings of the Symposium on High Performance Distributed Computing*, 1999, pp. 3–6.

[96] C. Guangzhu, L. Zhishu, Y. Daohua and Nimazhashi, 'A Model of Multi-Agent System Based on Immune Evolution', in *19th International Conference on Advanced Information Networking and Applications (AINA'05) Volume 1 (AINA papers)*, vol. 1, IEEE, pp. 53–58.

[97] L. Baresi, S. Guinea and L. Pasquale, 'Self-healing BPEL processes with Dynamo and the JBoss rule engine', in *International workshop on Engineering of software services for pervasive environments in conjunction with the 6th ESEC/FSE joint meeting - ESSPE '07*, New York, New York, USA: ACM Press, Sep. 2007, pp. 11–20.

[98] S. Guinea, 'Dynamo: A Framework for the Supervision of Web Service Compositions', PhD thesis, Politecnico di Milano – Dipartimento di Elettronica e Informazione, 2007.

[99] D. A. Menascé, M. N. Bennani and H. Ruan, 'On the Use of Online Analytic Performance Models, in Self-Managing and Self-Organizing Computer Systems', *Selfstar Properties in Complex Information Systems*, Lecture Notes in Computer Science, vol. 3460, O. Babaoglu, M. Jelasity, A. Montresor, C. Fetzer, S. Leonardi, A. Moorsel and M. Steen, Eds., pp. 128–142, 2005.

[100] V. Stantchev and C. Schröpfer, 'Negotiating and Enforcing QoS and SLAs in Grid and Cloud Computing', *Advances in Grid and Pervasive Computing*, pp. 25–35, 2009.

[101] A. Gehlert and A. Heuer, 'Towards Goal-Driven Self Optimisation of Service Based Applications', in *Towards a Service-Based Internet*, ser. Lecture Notes in Computer Science, P. Mähönen, K. Pohl and T. Priol, Eds., vol. 5377, Springer Berlin / Heidelberg, 2008, pp. 13–24.

[102] P. D'haeseleer, S. Forrest and P. Helman, *An immunological approach to change detection: algorithms, analysis and implications*. IEEE Comput. Soc. Press, 1996, pp. 110–119.

[103] R. Sterritt and M. Hinchey, 'Biologically-Inspired Concepts for Autonomic Self-Protection in Multiagent Systems', *Safety and Security in Multiagent Systems*, pp. 330–341, 2009.

[104] G. Weiss, D. Eilers and M. Zeller, *Towards automotive embedded systems with self-x properties*. INTECH Open Access Publisher, 2011.

[105] R. Michalski, J. Carbonell and T. Mitchell, *Machine Learning: An Artificial Intelligence Approach*, ser. Symbolic Computation. Springer-Verlag Berlin Heidelberg, 1983.

[106] J. R. Quinlan, *C4.5: Programs for Machine Learning*, ser. Morgan Kaufmann series in Machine Learning 3. Morgan Kaufmann, 1993, vol. 1, p. 302.

[107] J. R. Quinlan, 'Induction of decision trees', *Machine learning*, vol. 1, no. 1, pp. 81–106, 1986.

[108] J. Hadden, A. Tiwari, R. Roy and D. Ruta, 'Churn prediction: Does technology matter', *International Journal of Intelligent Technology*, vol. 1, no. 2, pp. 104–110, 2006.

[109] W. W. Cohen, 'Fast Effective Rule Induction', in *Proceedings of the Twelfth International Conference on Machine Learning*, A. Prieditis and S. Russell, Eds., Morgan Kaufmann, vol. 3, Morgan Kaufmann, 1995, pp. 115–123.

[110] P. K. Chan and S. J. Stolfo, 'Toward scalable learning with non-uniform class and cost distributions: A case study in credit card fraud detection.', in *KDD*, vol. 1998, 1998, pp. 164–168.

[111] P. Domingos, 'MetaCost: A General Method for Making Classifiers Cost-Sensitive', in *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, ACM, 1999, pp. 155–164.

[112] J. R. Koza, *Genetic programming: On the programming of computers by means of natural selection*. MIT press, 1992, vol. 1.

[113] L. de Castro and J. Timmis, *Artificial Immune Systems: A New Computational Intelligence Approach*. Springer, 2002.

[114] A. Watkins and J. Timmis, ‘Exploiting parallelism inherent in AIRS, an artificial immune classifier’, in *Artificial Immune Systems*, G. Nicosia, V. Cutello, P. J. Bentley and J. Timmis, Eds., ser. Lecture Notes in Computer Science, vol. 3239, Springer, 2004, pp. 427–438.

[115] L. N. D. Castro and F. J. V. Zuben, ‘Learning and optimization using the clonal selection principle’, in *IEEE Transactions on Evolutionary Computation*, vol. 6, IEEE, 2002, pp. 239–251.

[116] J. Brownlee, ‘Clonal Selection Theory & CLONALG - The Clonal Selection Classification Algorithm (CSCA)’, Centre for Intelligent Systems, Complex Processes (CISCP), Faculty of Information and Communication Technologies (ICT), Swinburne University of Technology, Victoria, Australia, Tech. Rep., 2005.

[117] J. H. Carter, ‘The immune system as a model for pattern recognition and classification.’, *Journal of the American Medical Informatics Association*, vol. 7, no. 1, pp. 28–41, 2000.

[118] D. Tay, C. Poh and R. Kitney, ‘An evolutionary data-conscious artificial immune recognition system’, in *Genetic and Evolutionary Computation Conference, GECCO ’13, Amsterdam, The Netherlands, July 6-10, 2013*, 2013, pp. 1101–1108.

[119] S. A. M. Elsayed, S. Rajasekaran and R. A. Ammar, ‘AC-CS: an immune-inspired associative classification algorithm’, in *Artificial Immune Systems - 11th International Conference, ICARIS 2012, Taormina, Italy, August 28-31, 2012. Proceedings*, 2012, pp. 139–151.

[120] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann and I. H. Witten, ‘The WEKA data mining software: an update’, *SIGKDD Explorations*, vol. 11, no. 1, pp. 10–18, 2009.

[121] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Artifical Selection*. Cambridge, MA, USA: MIT Press, 1992.

[122] A. E. Eiben and S. K. Smit, ‘Evolutionary algorithm parameters and methods to tune them’, in *Autonomous Search*, Springer, 2012, pp. 15–36.

[123] B. Calvez and G. Hutzler, ‘Automatic tuning of agent-based models using genetic algorithms’, *Multi-agent-based simulation VI*, 2006.

[124] R. G. Sargent, ‘Verification and validation of simulation models’, in *Proceedings of the 37th Conference on Winter Simulation*, ser. WSC ’05, Orlando, Florida: Winter Simulation Conference, Dec. 2005, pp. 130–143.

[125] K. Alden, M. Read, J. Timmis, P. S. Andrews, H. Veiga-Fernandes and M. Coles, ‘Spartan: A comprehensive tool for understanding uncertainty in simulations of biological systems’, *PLOS Computational Biology*, vol. 9, no. 2, e1002916, 2013.