Meta-data to Enhance Case-Based Prediction

A thesis submitted as partial fulfilment of the requirements
for the degree of

Doctor of Philosophy

November 2006

Rahul Premraj
Bournemouth University
School of Design, Engineering and Computing

Page Numbering as Bound

Meta-data to Enhance Case–Based Prediction

Rahul Premraj

Abstract

The focus of this thesis is to measure the regularity of case bases used in Case-Based Prediction (CBP) systems and the reliability of their constituent cases prior to the system's deployment to influence user confidence on the delivered solutions. The reliability information, referred to as meta-data, is then used to enhance prediction accuracy.

CBP is a strain of Case-Based Reasoning (CBR) that differs from the latter only in the solution feature which is a continuous value. Several factors make implementing such systems for prediction domains a challenge. Typically, the problem and solution spaces are unbounded in prediction problems that make it difficult to determine the portions of the domain represented by the case base. In addition, such problem domains often exhibit complex and poorly understood interactions between features and contain noise. As a result, the overall regularity in the case base is distorted which poses a hindrance to delivery of good quality solutions.

Hence in this research, techniques have been presented that address the issue of irregularity in case bases with an objective to increase prediction accuracy of solutions. Although, several techniques have been proposed in the CBR literature to deal with irregular case bases, they are inapplicable to CBP problems. As an alternative, this research proposes the generation of relevant case-specific meta-data. The meta-data is made use of in Mantel's randomisation test to objectively measure regularity in the case base. Several novel visualisations using the meta-data have been presented to observe the degree of regularity and help identify suspect unreliable cases whose reuse may very likely yield poor solutions. Further, performances of individual cases are recorded to judge their reliability, which is reflected upon before selecting them for reuse along with their distance from the problem case. The intention is to overlook unreliable cases in favour of relatively distant yet more reliable ones for reuse to enhance prediction accuracy.

Abstract

The proposed techniques have been demonstrated on software engineering data sets where the aim is to predict the duration of a software project on the basis of past completed projects recorded in the case base. Software engineering is a human-centric, volatile and dynamic discipline where many unrecorded factors influence productivity. This degrades the regularity in case bases where cases are disproportionably spread out in the problem and solution spaces resulting in erratic prediction quality.

Results from administering the proposed techniques were helpful to gain insight into the three software engineering data sets used in this analysis. The Mantel's test was very effective at measuring overall regularity within a case base, while the visualisations were learnt to be variably valuable depending upon the size of the data set. Most importantly, the proposed case discrimination system, that intended to reuse only reliable similar cases, was successful at increasing prediction accuracy for all three data sets.

Thus, the contributions of this research are some novel approaches making use of meta-data to firstly provide the means to assess and visualise irregularities in case bases and cases from prediction domains and secondly, provide a method to identify unreliable cases to avoid their reuse in favour to more reliable cases to enhance overall prediction accuracy.

Acknowledgements

First and foremost, I wish to thank my supervisor Prof. Martin Shepperd profusely for his support and unshaken faith in me throughout the course of this research. Over the last three years, Martin has worn several hats including that of a supervisor, a mentor and a friend. The thesis would not have reached this stage without his unlimited patience and encouragement. Thanks a tonne Martin. I owe you a lot for this!!!

I also express my gratitude to my second and third supervisors, Dr. Keith Phalp and Dr. Michelle Cartwright respectively for their periodic feedback on my work and the thesis. A special thanks also goes to Dr. Thomas Roth-Berghofer who gladly shared his elegant PhD LaTeX template to write my thesis.

Next in queue are the numerous friends (you know who you are ;-)) whose best wishes and support went a long way towards keeping me on track. A special thanks to Chris Pauli for being an equal partner in the quest of driving the take-away food industry in Bournemouth. Good luck for your thesis! Also, many thanks to John Kanyaru for comforting me when I was lost by saying I am not alone. Good luck with your PhD viva!

Last and most importantly, I thank my parents, C. Premraj and Sunita Premraj and my sister Richa Sidhu for their endless love and support while I have been far away from home. I am indebted to them for having left no stone unturned to provide me the means to ensure that I pursue an education and career of my choice.

Rahul Premraj

Bournemouth, UK

January 23, 2007

List of Publications

► **2006**

▷ Schröter, A. Zimmermann, T. **Premraj, R.** and Zeller, A. *"Where Do Bugs Come From?"*, In Procs. of the 14^{th} ACM SIGSOFT Symposium on Foundations of Software Engineering (Poster Paper), Portland, 2006.

▷ Schröter, A. Zimmermann, T. **Premraj, R.** and Zeller, A. *"If Your Bug Data Base Could Talk..."*, In Procs. of the 5^{th} ACM/IEEE International Symposium on Empirical Software Engineering (Short Paper), Rio de Janeiro, 2006.

► **2005**

▷ **Premraj, R.** and Shepperd, M. *"Assessing Case Base Quality"*, In Procs. of UK CBR Workshop, Cambridge, 2005.

▷ **Premraj, R.** Shepperd, M. Kitchenham B. and Forselius, P. *"An Empirical Analysis of Software Productivity Over Time"*, In Procs. of the 11^{th} IEEE International Software Metrics Symposium, Como, Italy, 2005.

► **2004**

▷ **Premraj, R.** Twala, B. Mair, C. and Forselius, P. *"Productivity of Software Projects by Business Sector: An Empirical Analysis of Trends"*, In Procs. of the 10^{th} IEEE International Software Metrics Symposium, Chicago, USA, 2004.

▷ **Premraj, R.** Shepperd, M. and Cartwright, M. *"Data Enrichment in Case-Based Reasoning for Software Cost Prediction"*, In Procs. of PREP2004, IEEE, Hatfield UK, 2004.

► **2003**

▷ **Premraj, R.** Shepperd, M. and Cartwright, M. *"Meta-Data to Guide Retrieval in CBR for Software Cost Prediction"*, In Procs. of UK CBR Workshop, Cambridge, 2003.

- Mair, C., Shepperd, M Cartwright, M. Kirsopp, C. **Premraj, R.** and Heathcote, D. *"Understanding Object Feature Binding Through Experimentation and Modelling"* In Procs. of the 8^{th} Neural Computation and Psychology Workshop, Connectionist Models of Cognition and Perception II, Boston, USA, 2003.
- Kirsopp, C. Mendes, E. **Premraj, R.** and Shepperd, M.J. *"An Empirical Analysis of Linear Adaptation Techniques for Case-Based Prediction"*, In Procs. of the 5^{th} International Conference on Case–Based Reasoning, pp231–245, Trondheim, Norway: Springer-Verlag, 2003.

Contents

Abstract v

Acknowledgements vii

List of Publications ix

I Background 1

1 Introduction 3

1.1 Software Engineering Estimation 4

1.2 Motivation for Thesis 5

1.2.1 Software Engineering Perspective 5

1.2.2 Case–Based Maintenance Perspective 6

1.3 Research Aims and Objectives 6

1.4 Thesis Map 7

2 Case Based Reasoning and Prediction 9

2.1 CBR: Origin and Mechanics 9

2.2 Advantages of the CBR Approach 11

2.3 Disadvantages of the CBR Approach 12

2.4 Case–Based Prediction 13

2.5 Summary 15

3 Software Engineering Estimation 17

3.1 Algorithmic Methods 17

3.2 Expert Judgement 18

3.3 Machine Learning Methods 19

3.4 CBR in Software Engineering Estimation 20

3.5 Chapter Summary 23

4 Case Base Maintenance **25**

4.1 Maintenance in Case–Based Reasoning 25

4.2 CBM in the CBR Cycle 27

4.3 Types of CBM Techniques 28

4.4 Case Base Editing Techniques 29

4.4.1 Competence Preserving 29

4.4.2 Competence Enhancing 32

4.5 Discussion 33

II Methodology **37**

5 Meta-Data **39**

5.1 Meta-Data in CBR 39

5.2 Desharnais Data Set 40

5.3 Distance and Residual Rank Ratios 41

5.3.1 Distance Rank Ratio 42

5.3.2 Residual Rank Ratio 42

5.4 Generation of Meta-Data 44

5.5 Chapter Summary 46

6 Case Base and Case Quality **47**

6.1 Mantel Randomisation Test 47

6.2 Visualising Case Base Quality 50

6.2.1 Overall Case Base Dissonance 51

6.2.2 Case Specific Dissonance 52

6.2.3 Spearman's Correlation 54

6.3 Case Profile 57

6.4 Chapter Summary 60

7 Enhancing Case–Based Prediction **63**

7.1 k-NN 63

7.2 Frequentist Approach for Case Assessment 64

7.3 Procedure 66

7.4 Analysis of Results 66

7.5 Chapter Summary 67

III Results and Discussion 69

8 Data Sets 71
8.1 British Telecom Data Set 71
8.2 Desharnais Data Set 72
8.3 Finnish Data Set 72
8.4 Chapter Summary 74

9 Results 77
9.1 BT Data Set 77
9.1.1 Mantel Randomisation Test 77
9.1.2 Visualisation Case Base Quality 79
9.1.3 Case Profile 83
9.1.4 Enhancing Prediction 84
9.2 Desharnais Data Set 87
9.2.1 Mantel's Randomisation Test 87
9.2.2 Visualisation Case Base Quality 90
9.2.3 Case Profile 93
9.2.4 Prediction 95
9.3 Finnish Data Set 100
9.3.1 Mantel Test 100
9.3.2 Visualisation Case Base Quality 101
9.3.3 Case Profile 108
9.3.4 Prediction 110
9.4 Retrieval Demonstrations 113
9.5 Discussion 115

IV Conclusions 119

10 Conclusions 121
10.1 Summary of Research 121
10.2 Research Objectives 122
10.3 Summary of Research Findings 124
10.4 Contributions of the Thesis 125
10.5 Limitations of Work 126
10.5.1 Limitations of Approach 126

10.5.2 Tool Limitations 126
10.5.3 Analysis Limitations 127
10.6 Future Extensions to Research 128
10.6.1 Tool Support 128
10.6.2 Method Refinement and Enhancement 129
10.6.3 General Applicability 130
10.6.4 Software Engineering Estimation 130
10.7 Summary 130

V Appendices 131

A Script Listing - Main.m 133
A.1 Main Script 133
A.2 Splitting Data sets into Training and Testing Samples 144
A.3 Normalising Distances 145
A.4 Calculating Inter-Case Distance 146
A.5 Sorting Matrices 147
A.6 Calculating Mantel's Correlation 147
A.7 Calculating Probability from Case Profile 148
A.8 Predicting Solutions 151

VI Bibliography 153

Bibliography 155

List of Figures

2.1 Aamodt and Plaza's CBR Cycle [AP94] 10

4.1 R6 Model [RBI01] 28

6.1 Distance Rank Ratio$_{C,T}$ Vs. Residual Rank Ratio$_{C,T}$ 52
6.2 Sorted $DRR_{C,T}$ Matrix 53
6.3 $RRR_{C,T}$ Matrix sorted by $DRR_{C,T}$ 53
6.4 $RRR_{C,T}$ Matrix sorted by $DRR_{C,T}$ 56
6.5 A Case Profile Example 58
6.6 Reliable Case Example 59
6.7 Reliable Case Example 59
6.8 Unreliable Case Example 59
6.9 Unreliable Case Example 59

9.1 $DRR_{C,T}$ vs. $RRR_{C,T}$ for Regular BT Sample 6 79
9.2 $DRR_{C,T}$ vs. $RRR_{C,T}$ for Irregular BT Sample 10 79
9.3 $RRR_{C,T}$ sorted by $DRR_{C,T}$ for Regular BT Sample 6 81
9.4 $RRR_{C,T}$ sorted by $DRR_{C,T}$ for Irregular BT Sample 10 81
9.5 Case Profile for Reliable `Case 7` from `BT Sample 6` 84
9.6 Case Profile for Reliable `Case 7` from `BT Sample 6` 84
9.7 Case Profile for Unreliable `Case 4` from `BT Sample 10` 85
9.8 Case Profile for Unreliable `Case 4` from `BT Sample 10` 85
9.9 Mean of $Sum|Res|$ from BT Data Set 86
9.10 $DRR_{C,T}$ vs. $RRR_{C,T}$ for Regular Desharnais Sample 5 88
9.11 $DRR_{C,T}$ vs. $RRR_{C,T}$ for Irregular Desharnais Sample 14 88
9.12 $RRR_{C,T}$ sorted by $DRR_{C,T}$ for Regular Desharnais Sample 5 90
9.13 $RRR_{C,T}$ sorted by $DRR_{C,T}$ for Irregular Desharnais Sample 14 91
9.14 Case Profile for `Case 49` from `Desharnais Sample 5` 94
9.15 Case Profile for `Case 49` from `Desharnais Sample 5` 94

9.16 Case Profile for `Case 39` from `Desharnais Sample 14` 94
9.17 Case Profile for `Case 39` from `Desharnais Sample 14` 94
9.18 Comparison of Performance (Means) by Coupling k-NN and Probability Threshold for the Desharnais Dataset 96
9.19 Comparison of Performance (Medians) by Coupling k-NN and Probability Threshold for the Desharnais Dataset 97
9.20 $DRR_{C,T}$ vs. $RRR_{C,T}$ for Finnish Sample 19 103
9.21 $DRR_{C,T}$ vs. $RRR_{C,T}$ for Finnish Sample 9 104
9.22 $RRR_{C,T}$ sorted by $DRR_{C,T}$ for Finnish Sample 19 107
9.23 $RRR_{C,T}$ sorted by $DRR_{C,T}$ for Finnish Sample 9 107
9.24 $RRR_{C,T}$ sorted by $DRR_{C,T}$ for Finnish Sample 19 (Enhanced) 108
9.25 Case Profile for Reliable `Case 110` from `Finnish Sample 19` 109
9.26 Case Profile for Reliable `Case 110` from `Finnish Sample 19` 109
9.27 Case Profile for Unreliable `Case 77` from `Finnish Sample 20` 109
9.28 Case Profile for Unreliable `Case 77` from `Finnish Sample 20` 109
9.29 Comparison of Performance (Means) by Coupling k-NN and Probability Threshold for the Finnish Dataset 110
9.30 An Example of Increase in Prediction Accuracy 113
9.31 An Example of Decrease in Prediction Accuracy 114
9.32 Case Profile of Rejected Case 115

List of Tables

3.1 Comparison of Different Software Engineering Estimation Models 22

5.1 Rank Ratio Example 45

6.1 Case Quality using Spearman's Rank Correlation 55

8.1 BT Data Set Feature List 72
8.2 Desharnais Data Set Feature List 73
8.3 Finnish Data Set Feature List 75

9.1 Mantel's Randomisation Test Results on the BT Data Set 80
9.2 Case Quality using Spearman's Rank Correlation 82
9.3 Mean of *Sum*|*Res*| from the BT Data Set 86
9.4 Mantel's Randomisation Test Results on the Desharnais Data Set 89
9.5 Case Quality using Spearman's Rank Correlation for `Desharnais Sample 5` 91
9.6 Case Quality using Spearman's Rank Correlation for `Desharnais Sample 14` 92
9.7 *Sum*|*Res*| from 30 Random Samples of the Desharnais Data Set 98
9.8 Kruskal-Wallis Test Results for Desharnais Data Set 99
9.9 Mantel's Randomisation Test Results on the Finnish Data Set 102
9.10 Case Quality using Spearman's Rank Correlation for `Finnish Sample 19` 105
9.11 Case Quality using Spearman's Rank Correlation for `Finnish Sample 9` 106
9.12 Mean of *Sum*|*Res*| from 30 Random Samples of the Finnish Data Set . . 111
9.13 Kruskal-Wallis Test Results for Finnish Data Set 112
9.14 Summary of Effectiveness of Techniques 116

1

Background

PAGE

MISSING

IN

ORIGINAL

CHAPTER 1

Introduction

Case-Based Reasoning (CBR) [AP94] is a relatively young, yet vibrant and popular Artificial Intelligence (AI) technique. Over the last 15 years, researchers have shown considerable interest in CBR by further maturing the approach via research and successfully applying it to many diverse applications. Its widespread adoption is much attributed to its deviance from traditional problem-solving approaches in computing.

The CBR methodology is specially geared towards knowledge deficient domains and is merited for its flexibility and applicability to a rich variety of tasks [Aha98]. It mimics the problem-solving process of human experts, who use their experience accumulated over the years as starting points to solve new problems. For example, an experienced engineer may fix a computer by recalling a past job where the previous computer exhibited similar symptoms believing it is highly likely that the same solution may work again.

Similarly, CBR systems are endowed with a case base(s) which is a repository of domain-specific instances of previously solved problems, each referred to as a case. Faced with a new problem (target case) to solve, the system scans its case base to find the most similar case using a suitable distance metric. The solution to the retrieved similar case is then reused to derive one for the target case. Hence, the successful operation of the entire system is crucially dependent on the premise – *"Similar problems have similar solutions"* [Kol93]. This is elaborated upon further in Chapter 2. CBR systems heavily rely upon their case base that is the basis of every problem-solving exercise. Thus, the quality or competence of a case base is crucial and can be judged on the basis of many parameters including its representation or coverage of the problem space, inherent noise, redundancy, efficiency and so on. One such important parameter for judgement is problem-solution regularity [LW99] which is a central and

critical phenomenon for delivery of good solutions. Regularity in a case base measures the degree to which cases lie within proportional distances from each other in both problem and solution spaces. Case bases with more inherent regularity are likely to deliver quick and reliable solutions. On the other hand, there are slimmer chances of obtaining equally desirable solutions when using a relatively inferior or irregular case base.

This research focusses upon the aspect of dealing with irregular case bases in a particular strain of CBR commonly referred to as Case–Based Prediction (CBP) in which the solution is a continuous value. To do so, the domain of software project effort prediction is used (Section 1.1), where the objective is to accurately predict the effort of new projects based on similar existing and completed projects in the case base. Whilst this particular application of CBP has attracted a substantial amount of research interest, a problem has been the somewhat erratic results in terms of prediction quality.

1.1 Software Engineering Estimation

One class of prediction problem that has had some success applying case-based reasonering is software project effort estimation*. Early estimation of software projects is commercially important — since effort is generally the dominant component of cost — but in many respects an extremely challenging problem domain. Problems include small, noisy, heterogeneous and incomplete data sets coupled with large sets of categorical and continuous features that typically exhibit complex interactions [MSJ05]. In addition, the solution feature is a continuous value which makes it hard to measure, judge or even compare accuracy. Nonetheless early work, e.g. [PVM96, SS97] produced encouraging results and outperformed traditional methods such as stepwise regression analysis.

But despite ongoing progress, results have not been consistent among research groups or even among different random holdout sets. Mair and Shepperd [MS05] conducted a systematic review of published empirical studies using case-based prediction for project effort. Twenty distinct studies were identified that compared CBR and some form of regression analysis. Of these 9 supported case-based prediction, 7

*In this thesis, the terms estimation and prediction will be used interchangeably as this is common practice in the field.

regression analysis and 4 were inconclusive. Further analysis reveals that one source of variation is the data sets used as case bases (also see [KCCS00]). For this reason, it was decided to investigate further and in particular into problem–solution irregularity, where for example, projects that are close neighbours in the feature space but possess strongly divergent solutions (which, in this research is project effort). This domain is discussed in detail in Chapter 3.

1.2 Motivation for Thesis

The motivations to undertake this research are now addressed from the perspectives of both, the application domain (software engineering estimation) as well as the adopted methodology (CBP).

1.2.1 Software Engineering Perspective

- A vast amount of money is spent on developing software worldwide. The successful execution of such projects is dependent upon accurate cost estimated from the start. Thus, it is crucial that work in software effort estimation is continued to further refine existing techniques for the benefit of the industry.
- Software development is a highly human centric problem. Hence, differences in productivity appear due to external variation, which may be unaccounted for by the feature set characterising the project state [Sca94, Arm02, JIS03]. Such variation may inject an inevitable degree of randomness into the project that distorts overall problem-solution regularity in the case base and results in poor solution accuracy.

  Thus, firstly it is important to verify if there is any value in using the data sets at all depending upon the degree of noise or irregularity. Secondly, it is valuable to have techniques in place that address this issue of noise or irregularity in the data sets to improve solution accuracy.
- It may be argued that incorporating sophisticated adaptation routines can help in obtaining more desirable solutions. Unfortunately the software engineering experts do not completely comprehend the complexity of multiple confounding factors that together determine total cost or effort. Thus, at the current state, the

community's understanding of the domain is inadequate to implement adaptation techniques, which may be no more than induced rules from individual data sets. These would be further inapplicable since they may be severely localised due to heterogeneity across data sets.

1.2.2 Case–Based Maintenance Perspective

- To the best of the author's knowledge, very little work (e.g. [CP04]) has been undertaken to develop case base maintenance techniques for prediction problems. Most available techniques have been developed largely to be applied to analytic tasks [VO96] (e.g. classification, diagnosis, decision support). These are often ineffective when applied to synthetic problems such as prediction, design, planning and configuration since it is hard to measure the accuracy of solutions in such domains.

- Also, the utility problem [FR93] (i.e. unchecked increase in the size of case base reduces efficiency) has attracted more attention to controlling the size of the case base without compromising the solution accuracy for efficiency gains. Unfortunately, this does not account for dealing with noisy cases present in the case base.

Hence, there is a realisation that CBR systems need to deal effectively with noisy software engineering data sets to enhance solution accuracy. However, techniques that can be applied successfully for such prediction problems are currently lacking. Hence, there is a genuine need for the development of alternative mechanisms that address this issue and are more generically applicable.

1.3 Research Aims and Objectives

The aim of this research is to develop and evaluate techniques that enable assessment of the regularity in a case base prior to deployment to influence user confidence on the solution and thereafter, identify potentially unreliable cases and discriminate their use in favour of relatively distant but regular cases to enhance solution accuracy.

The work described in this thesis develops and validates an alternative technique that can be applied across a variety of CBR systems independent of the application type. The research aim is planned to be achieved by accomplishing the following objectives:

- **OB1:** To establish that there is a need for an alternative technique that addresses problem–solution irregularity in CBP domains.
- **OB2:** To assess the problem-solution regularity in a case base to determine its applicability to CBP and influence user confidence.
- **OB3:** To identify unreliable cases that distort the overall regularity of a case base.
- **OB4:** To investigate if case reliability can gainfully supplement inter-case distance measures to increase solution accuracy in CBP using the domain of software engineering estimation as an example.

1.4 Thesis Map

In this section, a guide to the structure of the thesis is presented. Note that relevant chapters are mapped to the objectives above via the respective keys.

Chapter 2 This chapter provides a general overview of the CBR methodology. It then elaborates upon the methodology's characteristics that make it suitable for application to certain types of domains.

Chapter 3 The focus in this chapter is on software cost estimation. Here, some history of cost estimation is provided along with references to select relevant models developed earlier. Thereafter, the nature of software engineering data sets and its suitability to CBP is presented along with supporting empirical evidence from previous research.

Chapter 4 Here a literature review of techniques pertaining to CBM is presented. This chapter argues and exhibits that research to date has largely focussed on analytic tasks. In parallel, it also demonstrates that these techniques may not be effectively applied to CBP *(OB1)*.

Chapters 5, 6 & 7 In these chapters, the methodology of the technique proposed to be used to assess case base quality (objectively and visually), identify individual un-

reliable cases and thereafter, use the gathered information to increase solution accuracy is presented *(OB2 & OB3)*.

Chapter 8 An introduction and a brief statistical summary of all data sets used in this research is presented.

Chapter 9 Results from implementation of the technique presented in Chapters 5, 6 & 7 on each data set are presented and compared with conventional techniques used in CBP. This is followed by a summary of overall results *(OB4)*.

Chapter 10 Lastly, the contribution and significance of the presented techniques is discussed, followed by conclusions and suggested future work.

CHAPTER 2

Case Based Reasoning and Prediction

CBR is a prime example of lazy learning techniques [Aha97] where the real work is deferred until the actual time of problem solving, i.e. very little or no work is done offline. A CBR system remains dormant until presented with a problem to solve and thereafter, returns to its inactive state. This is in contrast to eager learning techniques such as neural networks and fuzzy systems that learn from training data to make generalisations about the problem before they attempt to solve a problem. While each of the two kinds of techniques has its own benefits, in this chapter, the former technique is concentrated upon since it is the focus of this research.

This chapter begins with a description of the origins and workings of CBR systems. The focus then shifts towards the advantages and disadvantages of its application to help judge its suitability for different problem domains. Lastly, some of the existing CBR applications are visited including those involving prediction.

2.1 CBR: Origin and Mechanics

The origin of CBR is largely attributed to Schank [Sch82] whose research into cognitive science led to its conception. The idea was borrowed from the observation that humans commonly react to situations at hand by remembering previous similar situations. Later, this concept was formalised into models for practical implementation, such as in [Kol93].

The CBR approach is based on two tenets [Lea96]. The first is that similar problems have similar solutions. Thus, previously solved problems would make a good starting

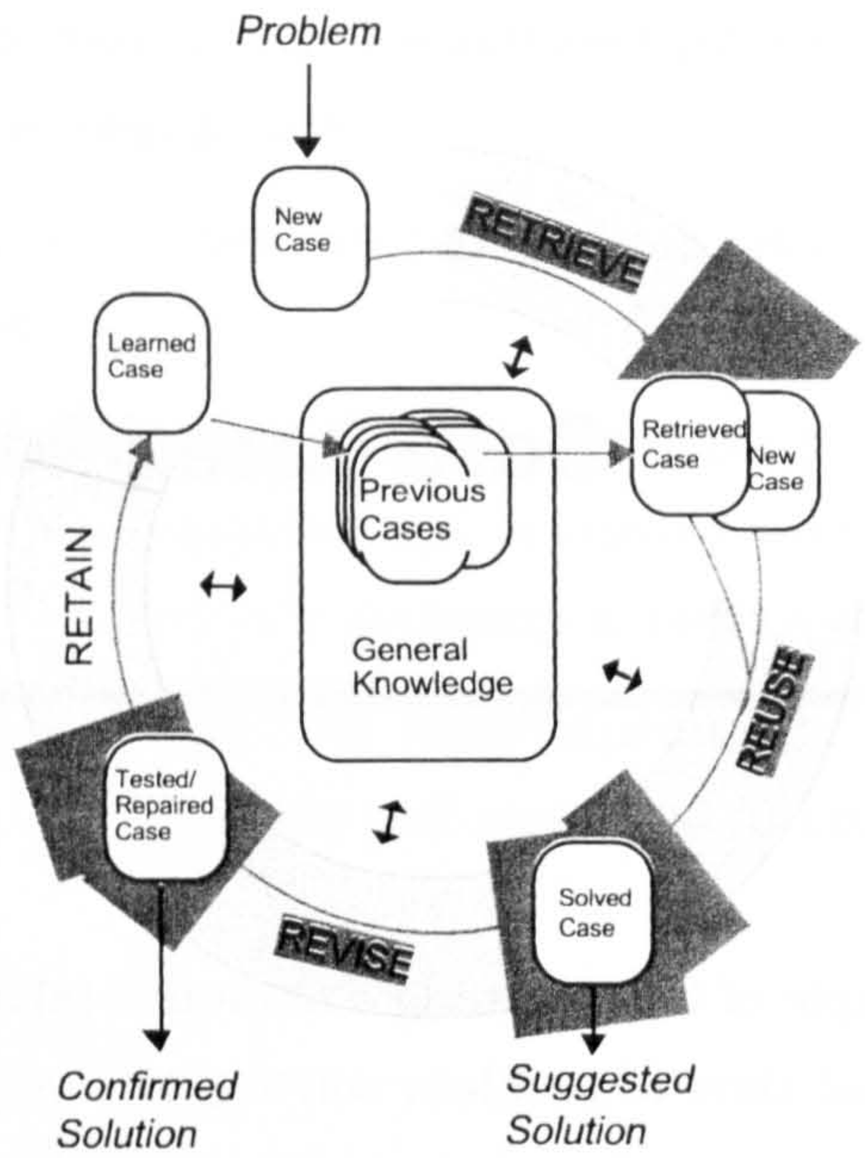

Figure 2.1: Aamodt and Plaza's CBR Cycle (AP94)

points to solve new similar solutions. The second tenet is that types of problems tend to recur within a domain. Hence, future problems within a domain are likely to be similar to those from the past.

With a relatively large research community focussing upon CBR in the last decade, its methodology has undergone a process of evolution over the years. Different models have been proposed, each contributing to a better understanding of the approach. Currently, Aamodt and Plaza's CBR model [AP94] (Fig. 2.1, see also [Alt89]) is widely accepted to be an inclusive representation of the methodology. This model is also popularly referred to as the R4 model since each of the four processes start with the same letter 'R'. The following four descriptions of each process are quoted from [AP94]:

Retrieve the most similar case or cases

Reuse the information and knowledge in that case to solve the problem

Revise the proposed solution

Retain the parts of this experience likely to be useful for future problem solving

Here, the broad idea is that on being presented with a new problem to solve, the system delves into its case base to search for similar previously solved problems. The relevant candidate (or most similar) case(s) is chosen objectively or contextually by a suitable distance measure. This characterises the retrieval stage. Thereafter, the solution or the process of its derivation is extracted in the Reuse stage and applied to the problem case along with general domain knowledge (if available) to propose a solution. The fitness of the solution is then tested and possibly repaired to adjust for non-corresponding features to be bettered in the Revise stage. Lastly, in the Retain stage a successfully solved problem, if considered useful for future use, is added to the case base with an intention to increase its quality or competence. Of course, each process poses several challenges to be correctly and efficiently applied for a given problem domain.

But widespread application of CBR has brought to light several deficiencies of the model and suggestions for modifications have been proposed. Cunningham [Cun98] pointed out that in practice, there is often a fuzzy distinction between the Reuse and Retain stages and suggested combining the two into a single Adaptation phase. Finnie and Wittig [FW03] suggested adding an additional stage, Repartition before Retrieve to be able to build case bases better by partitioning them into a world of problems and a world of solutions. While Reinartz *et al.* [RBI01] proposed two additional processes identifying that case base maintenance is crucial for the functioning of the entire system and must be explicitly represented in the model.

Nevertheless, there is a general consensus that the main aspects of the functioning of the CBR methodology are covered by Aamodt and Plaza's model which still serves as a platform to build upon for many researchers. It may be challenging to arrive at a more generalised model since different problem domains have distinct characteristics which largely determine the tasks and processes that need to be implemented.

2.2 Advantages of the CBR Approach

The CBR approach has several distinct advantages over conventional computing and AI techniques that make it attractive for application. To mention a few:

- CBR is especially favourable when applied to weak theory domains which are either poorly understood or lack algorithmic models. Derivation of solutions from previous experience makes it possible to overcome these problems.
- Due to reuse of previous experience, there is an impression that CBR may demand less knowledge engineering than alternative techniques [Cun98].
- On a wider scale, such systems can perform efficiently considering possibilities of centrally managed case bases that can be accessed by multiple users and also ensuring consistency amongst delivered solutions. This aspect may be very valuable in diagnostic CBR systems such as in the help desk domain [KB93].
- In some domains, it may be more efficient to derive competent solutions using analogy in comparison to use of first principle approaches. For example, under ideal circumstances in design and planning tasks, it may be easier to work upon an analogy to derive a solution rather than solving the problem from scratch.
- Unlike most other AI techniques, it is possible to reason, explain or even justify proposed solutions in a familiar way to increase user confidence, e.g [SCA05].
- CBR systems can be configured to learn over time. The knowledge learnt could be new cases, domain knowledge and adaptation routines through deduction and so on.

2.3 Disadvantages of the CBR Approach

Whilst CBR provides an array of advantages for use, it also has its limitations that may hinder its successful application or prove very challenging to implement in certain situations. These include:

- The CBR methodology was developed mainly for weak theory domains. Hence, it is likely that it may perform less favourably in domains that are well-defined and can potentially be modelled. Here, algorithmic models or rule-based systems may prove to be better alternatives.
- Finnie and Zhaohao [FS02] made an mathematical distinction between similarity and equivalence. Since CBR functions on the premise of similarity, which results in ballpark solutions, they are not suitable for application that require a great degree of precision.

- While CBR is merited for the advantages it provides on the knowledge engineering front, domains with fewer cases may have to rely on substantial general domain and adaptation knowledge. Gathering and implementation of such knowledge is a significantly challenging task.
- CBR may provide only a partial quick fix alternative to problems. Substantial research is involved in the successful application of the methodology such as choosing the most favourable case representation structure, similarity measures, etc. Additionally, its operation critically depends upon the existence of recorded episodes of problem-solving within a domain.

Thus, despite the benefits of applying CBR to characteristic problem domains, there also exist limitations that retard or constrain their implementation. For example, the effort saved on knowledge engineering may be compromised by consolidation of raw data as cases or development of adaptation routines to cater for sparsely covered areas in the problem space. Additionally, the quality of the solution depends upon the contents of the case base. Hence, it is crucial to endow the system with a case base that is capable of meeting expectations. Also, since each problem exhibits different characteristics, they need to be individually tackled to identify the most appropriate representation and processes to deliver effective and efficient solutions [Cun98].

2.4 Case-Based Prediction

The characteristics and advantages of CBR make it suitable to be applied to a variety of applications [Aha98, Alt01]. As mentioned in the previous chapter, Voss and Oxmon [VO96] classified CBR tasks or applications into analytic tasks and synthetic tasks. Applications belonging to the former class deal with analysing or interpreting the solution, e.g. classification tasks, diagnosis and decision support. While synthetic tasks cover those domains in which a solution needs to be composed or derived from other similar cases. Examples of such domains include design, planning and configuration.

In Case-Based Prediction (CBP), the solution is a continuous value. Hence, there is no certainty that every possible scenario in the domain is covered by the case base (this is also true for other synthetic domains, e.g. [Cra03]). As a result, an approximate solution is needed to be derived from one or more nearest case. The process adopted

for derivation of the solution is dependent upon the nature of the domain and its understanding. In some cases, this may be possible using adaptation techniques, while other knowledge deficient domains may resort to statistical learning approaches. This characteristic of CBP qualifies it as a synthetic task.

Besides the challenge of composing a continuous value solution, another problem lies in assessing the quality of solution. Unlike analytic problems in which the solution can be either correct or incorrect, synthetic tasks have to measure and infer the magnitude of error. In CBP, this translates into defining boundaries of good and poor solutions that are rather subjective. For example, can a solution with 10% error be regarded as good while another with 11% error regarded poor even though the magnitude of error is insignificant? Moreover, such judgement is made in light of the fact that the user expects an approximate solution. While this consideration is a short term problem, such statistics are cumulated to drive other processes in CBR in the longer term, e.g. maintenance of the case base by removal of redundant and noisy cases. Hence, even though delivering a solution in CBP may seem trivial, its implementation calls for application of techniques different to those of other domains due to the nature of the solution.

Still, CBP has been applied to several problems. Daengdej *et al.* [DLT+97, DLM99] used CBP for making car insurance claim predictions. Companies dealing in car insurance are required to factor in a 'predicted claims' component to determine the premium for a new customer. The idea was to seek claims made by previous customers and predict an averaged value for the new customer. But due to inconsistencies within their data set (90% of the cases had zero claims), simply using the nearest neighbour(s) for prediction was likely to give them an unacceptable solution. Hence, an hybrid-reasoning method that employed statistical methods including probability and regression was presented and shown to perform comparably with existing techniques used for prediction within the company.

Another example of CBP includes forecasting the sales of books [CL05] to minimize losses occurred by inaccurate demand estimation. Here, Chang and Lai constructed a hybrid system using CBP and self-organising maps and demonstrated the utility of assessing clusters of cases to better prediction accuracy. In [EA01], Essam and Ahmed applied CBP to predict the costs of constructing steel buildings to help in the bidding process. Their approach disintegrated each case into sub-cases and then retrieved

cases by measuring similarity between sub-problems. The final solution was delivered by combining or adapting the different solutions using neural networks.

Other examples of application of CBP include bankruptcy prediction [HI96], corporate bond rating prediction [KH01] and of course, software cost estimation [Sch98] which is the focal application of this research and is discussed in detail in the following chapter.

2.5 Summary

Over the years, CBR has been shown to be a viable and promising approach to cater to many problem domains where knowledge elicitation is a challenging task. The flexibility of its approach [Wat98] allows it to accommodate any relevant techniques suitable or specific to the problem domain.

Like other applications and domains, CBP too introduces a considerable number of its own challenges for implementation. Knowledge deficiency and a continuous value solution together call for more sophisticated techniques that specifically address problems encountered in prediction. One such problem is determining the quality of the solution given the ambiguity involved in its assessment and the expectation of being delivered an approximate outcome. The problem is augmented when dealing with irregular case bases, such as software engineering data sets. The research concentrates upon this aspect of CBP and proposes techniques that are relatively more objective and flexible to gauge solution quality, which further opens channels to other techniques that can be used to enhance solution quality. The next chapter presents an overview of software engineering estimation and highlight some of the issues faced with applying CBP to it.

CHAPTER 3

Software Engineering Estimation

Software engineers recognise that estimates of development effort play a very crucial role in the completion of projects. Not only are the estimates important to bid for tenders successfully, many financial decisions are made by management based upon such early estimates. While underestimates may lead to budget overruns and abandoning of the project, overestimates may cause commitment of too many resources to the project at the cost of other activities [LF02].

Due to its importance, over the past four decades, many attempts have been made to develop models for effort estimation. Several algorithmic and non-algorithmic models have been developed for this purpose. The following sections discuss some existing methods to give an overview of the state of affairs in software cost estimation.

3.1 Algorithmic Methods

Algorithmic models are statistical generalisations or parametric equations empirically derived from data sets that represent a relationship between cost or effort and other project characteristics. Thus, cost or effort estimate is considered dependent on other independent values such as lines of code, competence of staff and like.

Examples of such models include COCOMO [Boe81b], COCOMO II [BCHW95], Function Points [MBM94, NV97], SLIM [Put78] etc. Many researchers have comprehensively studied these models and compared their performances [MK92, Hee92, BW01, LF02]. But none of the models consistently performed well or even better than the others. Moreover, using some of the models resulted in errors of 100% or larger [Kem87]. Many

of the models have not been revised to keep pace with new technologies which has made them inapplicable to today's industry standards, e.g. Function Points may be inapplicable to today's world of new software technologies [Kit97]. Another issue is that these models are not straightforward to use. The models require considerable experience and competence to determine the feature values to input. Also, studies showed that the models need to be calibrated for use within organisations for better estimations [BES+99, Ves99, MK04]. This may be very discouraging for smaller software companies which may not have the competence or resources to adapt the models to suit their requirements.

3.2 Expert Judgement

Expert judgement involves consulting one or more domain experts. The human experts use their knowledge and experience to provide new estimates, where the experience relates to previous projects for which estimates may have been made in the past. Heemstra [Hee92] argues that the estimates provided by human experts are qualitative and not objective since they may be prone to bias, optimism, pessimism or instinct, desire to win or please [Del98] and cause under or overestimation without rational reasoning. Importantly, there is often a tendency in humans experts to forget [PS03]. Unsurprisingly, it has been observed that experts find it difficult to replicate or reproduce their results given the same parameters on which to base their judgement. This is more apparent when multiple experts are used to estimate a project and may arrive at non-concurring estimates. All these factors together make it extremely challenging to extrapolate the experts' basis for estimation and embed the knowledge constructively in estimation tools or techniques. However, despite the above short-comings of using human experts for estimation including the fact that such human experts are scarce and expensive to hire, a recent review by Moløkken and Jørgensen [MsJ03] revealed that amongst all estimation techniques, expert judgement is the most popular and preferred choice. Human experts are easy to employ in comparison to other techniques. Potentially the experts are updated with evolving new technology and hence are aware of compensating for relevant differences amongst different projects. Also, they have a psychological advantage since project managers may tend to trust a solution produced by a human expert. Some studies, including [Jør04], consider human

expert judgement to be a valuable technique such as the one conducted by Jeffery and Walkerden [JW99] to compare the performance of experts with and without the aid of tools. Results suggested that humans (without the aid of tools) were more competent at choosing relevant analogues from data sets and making appropriate estimations in comparison to automated tools. These findings were however contradicted in studies by Myrtveit and Stensrud [MS99].

3.3 Machine Learning Methods

The inadequacy of algorithmic techniques led researchers to experiment with non-parametric, especially machine learning (ML) alternatives for estimation. These techniques, including neural networks, analogy, fuzzy logic [MG96] and regression trees [SF95] were expected to learn the underlying relationship between features to deliver more accurate solutions. In this section, only the application of neural networks to estimation are discussed considering the the volume of research conducted, while estimation by analogy (CBR) is discussed in the following section.

One of the most widely experimented ML techniques for estimation are artificial neural networks (ANNs). Unfortunately, the focus has largely been upon comparing their performance against algorithmic techniques instead of further refining their implementation. An early promising attempt at using ANNs for estimation was by Venkatachalam [Ven93] who trained a multi-layer perceptron on the COCOMO data set using back-propagation. Later, Srinivasan and Fisher [SF95] used the same neural network architecture and trained the network on the COCOMO data set, yet tested it on the Kremerer data set. Their results showed ANNs to outperform the algorithmic models used in the analysis including COCOMO and SLIM. This was a considerably promising attempt since the training and testing data came from different data sets. Likewise, other researchers e.g. Hughes [Hug96] and Tadayon [Tad05] achieved good results by applying ANNs to software estimation.

However, the applications of ANNs also suffers from some drawbacks. Firstly, ANNs (such as multi-layer perceptrons) are known to be black boxes since it is not easy to extract the inherent relationships between features derived during training. Such knowledge may be important to gain acceptance of practitioners. Secondly, training

a neural network can be a very time consuming process since it involves choosing values by trial and error for several parameters such as the training function, the momentum and learning rate. Unfortunately, such parameters are also sensitive to small changes, making fining tuning the network all the more difficult. Despite these limitations, the results warrant further investigation into the use of this technique for estimation, although the focus needs to change to improve the configuration of the networks used.

3.4 CBR in Software Engineering Estimation

Several advantages of using CBR (Section 2.2) for software engineering estimation have propelled it to be one of the most intensively researched alternative to traditional and other contemporary techniques. Unlike neural networks, there seems to have been a balance between empirical validation of CBR for estimation purposes [FW97, FWD97, AS00] and refinement of the model.

The idea of using previous completed software projects to estimate effort of new projects was initially coined by Boehm [Boe81a]. Later, Vicinanza *et al.* [VPM90, VMP91] made one of the first attempts at using analogy for software cost estimation. Their tool Estor was similar to the contemporary CBR model and previously completed projects were made accessible. Similar projects were retrieved by finding sum of squares of differences and the solutions were adjusted using adaptation knowledge to account for non-corresponding features. Adjustments rules were in the form of if-then rules. e.g. if staff size of selected project is small and staff size of target project is large, then increase the effort estimation of target project by 20 percent. These rules were extracted from experts or were hand-coded to deliberately fit the data used, thus compromising their generality.

Another tool, FACE (Finding Analogies for Cost Estimation) was developed by Bisio and Malabocchia [BM95] which normalised candidate-target case similarity scores θ between 0 and 100 (100 being a perfect match). The value of θ was determined by the user and only those cases were reused to form an estimate whose similarity was beyond the threshold limit set.

Shepperd *et al.* developed a generic alternative to Estor for prediction purposes which was called ANGEL [SSK96, SS97, Sch98]. This tool allowed for searching similar projects in the case base using a variety of optional distance functions and then averaged them to deliver a solution. Later, Kadoda *et al.* [KCS00, KCCS00] conducted experiments on ANGEL to find optimal configurational parameter values for prediction such as number of nearest neighbours and similarity measure. However, the version of the tool allowed optimal feature set selecting using only exhaustive search which was computationally intractable for use once the number of features exceeded 15-20. Later, a revised version called ArchANGEL was developed that provided several feature and case subset selection strategies such as hill climbing methods [Ska94] and forward and backward sequential search. The use of these algorithms was shown to increase prediction accuracy [KS02a, KSH02]. Later, the group also explored other techniques to increase prediction accuracy including linear adaptation [KPS03] and avoiding use of misleading cases [PSC03].

Other initiatives to apply CBP for software estimation include work by Mendes *et al.* [MCM02, MCM03] whose interest lies in cost or effort estimation of web projects Delany *et al.* [DCW98, Del98, DC00] also investigated applicability of CBP to software estimation and argued that features included in software engineering data sets are inappropriate for estimation early on in the project's life-cycle. They made recommendations to collect data on subjective aspects of a project such as team experience, user requirements, requirements reliability and stability and suchlike. Idri and Abran [IA01, IAK02] recognised that often, software engineering data sets comprise of ordinal feature values and hence suggested the use of fuzzy logic to compute similarity between projects since it could handle linguistic values such as very low, low, high and very high.

A few studies have also been conducted comparing the performance of different techniques for software engineering estimation. Select ones have been presented in Table 3.1. The first column reports the relevant study and the remaining columns list the different prediction techniques used by each study. These include Case-Based Prediction, Advanced Case-Based Prediction (i.e. using some form of adaptation or feature subset selection in contract to a nearest neighbour approach), least square regression (LSR), rule induction (RI) and artificial neural networks (ANN). Each of these studies compared the performance of CBR with other techniques to investigate which of them delivers the best prediction accuracy. The maximum number of stars (*) on

Table 3.1: Comparison of Different Software Engineering Estimation Models

Study	CBP	Adv. CBP	LSR	RI	ANN
[SS97]	**		*		
[FW97]	**		*		
[BES+99]	*		**		
[MKL+00]	***		**	*	****
[SK01]	****		***	**	*
[KSH02]	*	**			
[MWT+03]	*		**		
[KPS03]	*	**			

each row indicate the number of techniques investigated in the study, and the most favoured technique is awarded those many stars. Thereafter, the second best technique is awarded a star lesser and so on and so forth.

The comparisons clearly show the potential of CBP for this domain. In the four studies ([SS97, FW97, MKL+00, SK01]), CBP outperformed LSR, but this trend was contradicted by the studies performed by Briand *et al.* [BES+99] and Mendes *et al.* [MWT+03]. Mair *et al.* [MKL+00] found ANNs to be the most accurate technique, but also emphasized that the achieved accuracy may not always outweigh the time spent on building and training complex models like ANNs. In contrast to them, Shepperd and Kadoda [SK01] found ANNs to be the poorest effort prediction technique when using simulated data. Whilst Table 3.1 lists only select studies, it is clear that there is some potential in the use of CBP for software engineering estimation in comparison to other techniques. This is further reinforced by the two studies conducted by Kirsopp *et al.* [KSH02, KPS03] where they found that advancing CBP by using adaptation rules and a better subset of features can further boost prediction accuracy of the technique.

Thus, on the whole the software engineering research community recognises the merits of applying CBP for estimation purposes. As a result, considerable groups are working on different techniques to further mature the approach while maintaining a balance with empirically validating the approach too.

3.5 Chapter Summary

Estimation of effort and cost is an important activity in the software engineering discipline. It has attracted large interest within the research community that has led to development of a variety of algorithmic and non-algorithmic models. While the former have shown to be inconsistent at delivering acceptable solutions and challenging at keeping abreast with current technologies, non-algorithmic models have attracted a large level of interest which led to the development of some promising alternatives.

CBP is one such widely researched and promising methodology. It has the advantages of being easily deployable, capable of explaination and importantly, adaptable to the constantly evolving software industry. But the erratic quality of solutions leaves ample room for betterment. As shown in [MS05], a cause for varying solution quality is the quality of data sets used as case bases and investigating the same is one of the foci of this research.

CHAPTER 4

Case Base Maintenance

Maintenance of a CBR system is crucial to facilitate continued achievement of predetermined objectives in the future [LW98]. This is a broad aspect in CBR which may involve activities such as checking the contents of the case base for their correctness or validity, fine-tuning retrieval strategies and making the system more efficient. Although this has long been realised, only recently has the community witnessed this aspect of CBR receiving the attention of researchers. As noted by Pal and Shiu [PS04], this flurry of activity has resulted in largely diverse pieces of work conducted under the absence of a unified framework. But lately, this has begun to change since alternative frameworks that can potentially serve as building blocks for pragmatic prototypes for maintenance are being proposed, for example [Wil01, RBR01, RB03]. In this chapter, broad issues pertaining to case base maintenance (CBM) including salient research that address some of them are visited. Later, in the discussion section, it is shown why the techniques are inapplicable for CBP problems and thus, reinforce the need for development of alternatives to cater for prediction problem case bases.

4.1 Maintenance in Case–Based Reasoning

CBR systems are widely accepted to comprise four knowledge containers [Ric98] *viz.* vocabulary, similarity measures, adaptation knowledge and the case base. While not every system may have all four knowledge containers, the contents are interchangeable between them. It is these knowledge containers upon which the system critically depends for problem–solving. But several factors affect the competence of their contents, such as obsolescence, redundancy and noise, and this degrades the overall perfor-

mance of the system. Thus, it is important to keep existing knowledge containers in shape to enable delivery of the best possible solutions.

A survey on case base maintenance by Iglezakis and Roth-Berghofer [IRB00] revealed that most research pertained to or concentrated upon the contents of the case base itself. This has resulted in the term case base maintenance being used synonymously with knowledge maintenance, although the latter is rather more generic since it covers all knowledge containers. But such predominant attention to the case base is unsurprising considering it is the crux of any CBR system. Also, a healthy or well-maintained case base facilitates solving a range of problems correctly and efficiently. This decreases dependence upon other knowledge containers (e.g. similarity measures and adaptation knowledge) for problem-solving, thereby reducing overall knowledge engineering effort.

Broadly speaking, CBM involves tasks or procedures implemented upon the CBR system to preserve or enhance its performance. Alternative definitions and descriptions of CBM have been proposed by different researchers and seem to be fairly consistent. One such generic definition by Leake and Wilson [LW98, Wil01] is as follows:

> *"Case-base maintenance implements policies for revising the organisation or contents (representation, domain content, accounting information, or implementation) of the case-base in order to facilitate future reasoning for a particular set of performance objectives."*

We now examine this definition more closely:

"Case-base maintenance implements policies": suggests that similar to other CBR sub-tasks, CBM is a methodology [Wat98] implemented by a set of guidelines rather than rules. Hence, CBM is undertaken keeping in mind the performance objectives or goals for a given system and domain.

"revising the organisation or contents": Revising the organisation refers to logical storage in the case base memory, while revising the contents refers to modifying the knowledge containers. It is important to note here that CBM extends to all knowledge containers in the case base i.e. including domain knowledge, similarity measures, vocabulary and adaptation knowledge. This is also in agreement with Zhu and Yang's classification of maintenance tasks [ZY99] which is discussed later in the chapter.

"a particular set of performance objectives": CBM may be implemented within a case-based reasoner for a variety of reasons. While the goal may be to only increase solution accuracy on one system, another system may need this goal to be accomplished using the smallest possible case base. Hence, CBM operations on the knowledge containers are implemented according to pre-determined goals and different goals could lead to quite different maintenance activities.

4.2 CBM in the CBR Cycle

An important aspect of CBM is its integration into the CBR-Cycle. In a broad sense, CBM has always been part of CBR systems. In Aamodt and Plaza's CBR cycle ([AP94] and Section 2.1), CBM is witnessed in the Retain stage (Fig. 2.1) where a new problem is added to the case base alongside its solution for future use. The notion behind adding a new case to the case base is to increase competence by enlarging its domain coverage for better problem solving in the future. Guided by a performance objective and resulting in the modification of a knowledge container qualifies the Retain step to be a CBM task, according to Leake and Wilson's definition presented in the previous section.

A major drawback of Aamodt and Plaza's model is the implicit inclusion of CBM, resultantly undermining its importance. This deficiency was recognised and corrected by Reinartz *et al.* [RBI01] who proposed a *R6 CBR-Cycle* which adds two additional steps *viz.* Review and Restore, to the R4 CBR cycle. The R6 CBR-Cycle is distinctly partitioned into Application Phase and Maintenance Phase (Fig. 4.1). While the Application phase focusses upon solving a given problem, the Maintenance Phase concentrates upon conditioning the case base to preserve its competence. This model is aimed at giving maintenance a more integrated, definite and functional role in CBR systems.

In the R6 model, the Review step entails assessing the status of various knowledge containers and monitoring them routinely, ad hoc or when triggered. The results of the assessment are thereafter reported indicating whether and which knowledge container may require maintenance. The Restore step short-lists and ranks the possible operations that may potentially be performed to reinstate the knowledge container to the desired level. This model is more elaborately discussed in [RBR01]. Besides, another

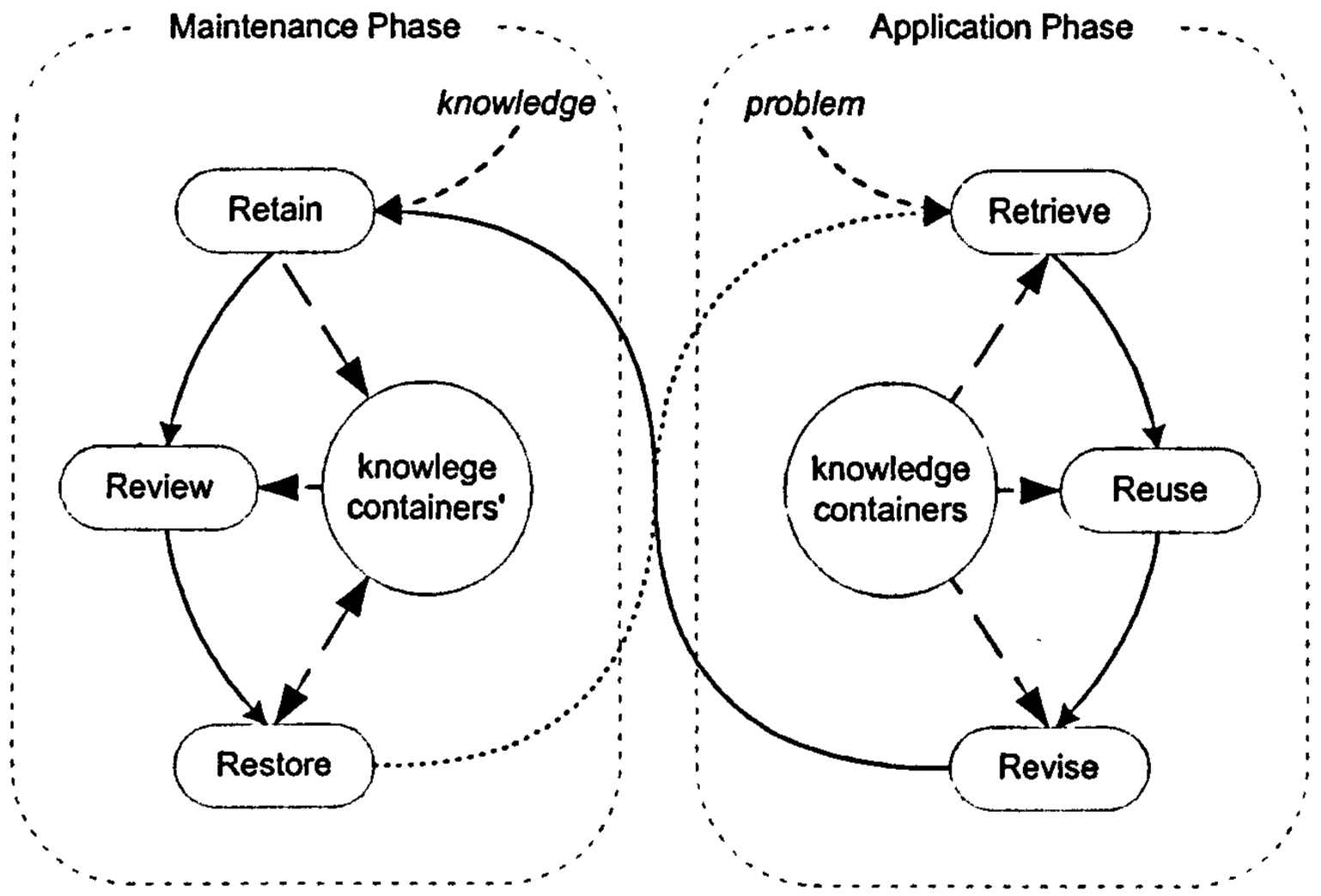

Figure 4.1: R6 Model (RBI01)

point of merit of the R6 model is that it accounts for the concepts of data collection and timing of implementing maintenance operations introduced by Leake and Wilson [LW98].

4.3 Types of CBM Techniques

Zhu and Yang [ZY99] suggested a target driven classification for CBM techniques. One class of techniques deal with contents of the knowledge containers in the system to enhance overall performance. As mentioned earlier, most such techniques focus upon the case base and tend to maintain or enhance its competence for problem-solving. Examples of such tasks include adding or deleting cases, redefining similarity measures and so on. Such maintenance tasks can be termed as Knowledge-Specific Maintenance (KSM). On the other hand, other techniques concentrate upon issues such as the representation of knowledge within the containers, case base indices, etc. Thus, they are aimed at structural changes in the system to enhance efficiency, e.g. [CBS97, ZY98]. Such tasks as can be classified as Structure-Specific Maintenance (SSM). It is important to note that these two classes are not necessarily exclusive since implementing one may result in or call for implementing the other.

Alternatively, Pal and Shiu [PS04] more recently suggested a method driven classification, i.e. Quantitative Maintenance or Qualitative Maintenance. As previously stated, a CBM is triggered or executed on a case base to accomplish pre-determined objectives. The former concentrates on efficiency related issues of CBR e.g. problem-solving efficiency, case base storage size, etc. while the latter deals with maintaining or enhancing the effectiveness of a system by addressing the case base's correctness, consistency and completeness. Similar to the above classification by Zhu and Zhang, both these categories are not necessarily exclusive since some objectives overlap and implementing one may result in or call for implementing the other.

4.4 Case Base Editing Techniques

Since this research centres around case base quality, Knowledge-specific maintenance techniques are focussed upon below. KSM or editing of case bases can be either decremental approaches, in which they reduce the size of the case base, or incremental approaches that add new cases to the case base. Brighton and Mellish [BM02] proposed another classification of approaches as competence preserving or competence enhancing techniques. The former class of techniques focuses upon superfluous instances whose removal does not lead to any decrease in classification accuracy, while in the latter class of techniques, noisy or possibly corrupt instances are identified and treated or removed to increase classification accuracy. We examine some such prominent techniques in the following two sub-sections.

4.4.1 Competence Preserving

Much recent research on CBM has been motivated by the utility problem [FR93], which in effect is competence preserving. In CBR, it is characterised by unchecked growth of knowledge (specifically cases) with an intention to improve performance, but results in quite the opposite i.e. degradation of overall performance, especially efficiency. This is because of the increase in size of the case base for systems that encounter a high frequency of problems. As a result, search and retrieval time increases with size and hampers quick solution delivery. This phenomenon of trade-off between solution quality and system efficiency, also referred to as the swamping problem [SG04], was

empirically validated by Smyth and Cunningham [SC96]. Thus, many techniques have been proposed to reduce the size of the case base and preserve its competence.

One of the earliest attempts was made by Hart who proposed the Condensed Nearest Neighbour (CNN) technique [Har68]. In CNN, cases are incrementally moved from the case base to an empty set only if their absence results in misclassification of cases in the edited set. As pointed out by Delany and Cunningham [DC04], this technique is sensitive to noise and the order of cases examined. CNN was further refined by Ritter [RWLI75] by accounting for adding noisy cases by adding only those new cases that are close in their problem and solution spaces to existing cases in the edited set. Again, this technique does not overcome the problem of case presentation order. A contrasting method to CNN was proposed by Gates [Gat72] who removed only those cases from the training set whose absence did not result in any misclassifications, but this is highly likely to be sensitive to the order of presented cases too. Aha *et al.*'s CBL1 learning algorithm [AKA91] was modified to develop CBL2 to selectively chose to retain only those cases that discriminated between goal feature values. Although it was shown that it considerably reduced the number of stored cases, it performed relatively poorly in comparison to CBL1 and was also vulnerable to noise.

Smyth *et al.* have published several research papers on maintaining case base competence while minimising size, the resultant case base being termed as competence footprint. Here, competence is defined as the area of the problem-space covered by the case base. In [SK95], Smyth and Keane introduced the concept of coverage and reachability. The coverage of a case is the set of target problems that can be solved, while the reachability of a target case is the set of cases that can solve it. On the basis of the degree of overlap between these two sets for individual cases, they were classified as pivotal, auxiliary, spanning and support cases. Then two case deletion policies were proposed, (a) the footprint deletion policy that deleted cases in the order – auxiliary, support, spanning and pivotal; and (b) footprint–utility deletion policy which deleted cases in the same order but in light of the case's utility. Thus, this technique removes only redundant cases from the case base which have no impact on the competence of the system, but retard efficiency. Empirical validation of the two policies on a residential property valuation system encouraged further work that was published in [SM98, SM99, MS00, SM01]. Several variants that borrowed fundamental concepts introduced by Smyth *et al.* have been proposed. For example, Brighton and Mellish [BM02] removed cases from the case base whose reachability set was larger than their

coverage set. Again, aiming to build a competent case base, Zhu and Yang [ZY99] presented an algorithm to contrarily add cases to the case base to maintain competence. Reusing the concepts of case coverage, their technique, testing upon a planning application, claimed to place a desirable lower bound on the competence of the resulting case base.

While the footprint technique was developed with an intent of removing redundant cases, it is important to note that the case deletion policies were based on two large assumptions that may fail to hold in case bases for prediction domains. Firstly, the algorithms assume that the case base is representative of the problem space [LW99] and secondly, the problem-space is a regular one. Also, no techniques have been suggested to verify if these assumptions hold for a given case base. In addition, competence preserving deletion strategies have shown to be effective, though on classification problems only. It is believed that these policies may be difficult to implement in domains where the solution is a continuous value since the concepts of coverage and reachability would be rather subjective.

Salamó and Golobardes [SG04] applied reinforcement learning (RL) to reduce the size or increase the competence of the case base. In their model, the RL algorithm continually monitors the state s_t of the system and awards a reward based on every performance. On failure, it triggers action to delete or maintain a case in the case base. A higher reward is awarded on problem–solving using the new reduced case base. On the whole, the important part of the of the RL algorithm is the score that reflects the on-going state of the CBR system. Thus, this model addressed the utility problem by correcting or removing noisy cases and has been evaluated using numerous classification data sets since it is clearer to recognise errors.

Another reason to remove cases from the case base is limitations on storage. Kira and Arkin [KA04] faced this problem due to fixed storage on mobile-robots who used CBR for navigation purposes. Here, the system is required to make room for a new case to be stored by deleting an existing case. Several metrics including recency, frequency of use, successes and random deletion were experimented with and were found to be effective.

There has also been research on maintenance of unstructured or textual case bases [RY96, RY97, RY01]. One of the foci was to detect and remove redundant cases in the case base. This was achieved by pre-processing problem and solution descriptions to

identify keywords and thereafter, comparing cases to observe the degree of overlap in both descriptions. Rules were presented to determine the nature of the overlap and selection of which case to remove.

4.4.2 Competence Enhancing

Competence enhancing techniques are those which are aimed at increasing the solution quality or accuracy of the CBR system. Such techniques are typically associated with noise removal which hamper the overall quality of solutions delivered. Noise in a case base can either be inherent or evolutionary. Inherent noise includes cases with inconsistencies or suspicious values in either, their problem or solution feature values. This could be a result of a false observation, a clerical error during data entry or may indeed be an outlier. On the other hand, evolutionary noise may be a result of changes in the domain which render older cases in the case base use lessor incorrect. Thus, in essence, the data is exhibiting a moving or evolving behaviour, e.g. behaviour of bank account holders over the past decade may exhibit differences rendering previous data unhelpful or even noisy.

An early attempt by Wilson [Wil72], also known as Wilson Editing, removed all cases in the case base which were incorrectly classified by their nearest neighbour. Though effective, a major drawback of this scheme was that the target case was assumed to be noise even though the candidate case may well have been so. Later, Tomek [Tom76] built upon this technique by making several passes over the training set and also by incrementing with different values of k (number of nearest neighbours).

From a case base maintenance perspective, CBL3 [Aha91] is of interest to us since it aimed at removing noisy cases. This was implemented by tracking the frequency of a retrieved case matching the target's goal feature. Thereafter, only those cases with significantly high frequencies were reused. Results showed that CBL3 was relatively more robust and resilient to noise in comparison to its predecessors.

A method that accounted for temporal drifts (evolutionary noise) was proposed by Montaner *et al.* [MLdeR02] for recommender agents. Their technique accounted for changing tastes and preferences of uses that made some existing cases irrelevant. Each case was associated with a variable that recorded the degree of users' interests in it. Upon reaching a certain threshold, it was discarded.

A failure driven deletion strategy was proposed by Portinale *et al.* [PTT99, PT00] where the entire case base was periodically monitored. Here, two types of cases were identified and removed. Firstly, cases that have been inactive or not retrieved for a threshold period of time are considered irrelevant and deleted. Secondly, cases that have been recorded to deliver false solutions more often are deleted.

An interesting extension to the footprint deletion policy [SM01] was proposed by Delany and Cunningham [DC04] who dealt with spam filtering which entailed using voluminous and noisy case bases. In addition to defining the coverage and reachability sets of a case, they included another property, i.e. the liability set which defines the instances where the case misclassifies the target case or contributes towards it. Thereafter, if cases within the coverage of this case can be classified correctly using other cases, it is deleted or else retained.

Cheetham [Che00] developed a technique that associated a confidence value with the proposed solution. The approach preprocessed the training data to assess the magnitude of error in the predicted value relative to the target-candidate case distance and generate a linear regression model between similarity scores and error. Although an interesting approach to influence users' confidence, this approach does not address the issue of avoiding reuse of noisy cases which principally affect the confidence intervals.

Thus, the competence enhancing techniques presented above deal with noisy cases in the case base that may pose a hurdle to delivery of good solutions.

4.5 Discussion

This chapter summarised recent salient developments in case base maintenance. As mentioned before, it is evident from the works cited that research in this aspect of CBR has gained momentum in the recent years. In this section, some of the characteristics and trends in maintenance relevant to this research are discussed. Also, certain gaps or inadequacies in the research are presented that make such techniques ineffective for prediction problems. Firstly, it was noted that most techniques were built for and demonstrated upon classification problems. Such tasks usually have well-defined problem and solution space boundaries that ease development of routines that perform intelligent operations such as maintenance. Additionally, classification case bases may

tend to exhibit more problem-solution regularity due to relatively lesser noise. This is because most boundary cases are problematic since they can potentially misclassify or be misclassified and hence, can be labelled as noise. Moreover, detecting noise in such case bases is relatively trivial since a case simply correctly or incorrectly classifies or is classified, irrespective of 'by how much'. This makes possible implementation of concepts such as coverage and reachability, as suggested by Smyth *et al.* [SK95], to be determined for cases in such case bases since the boundaries are more hardlined.

Whereas in software engineering estimation (and other prediction problem domains), even similar projects exhibit a certain degree of difference in the solution values since no two projects are likely to be identical in specifications and implementation. Additionally, every project is likely to be vulnerable to external factors that deviate the project from its schedule [Sca94]. Also, as discussed in Chapter 3, the problem and solution spaces in our domain are open-ended. Hence, it is nearly impossible to ascertain whether a given case base well represents the domain. This further makes the task of confidently defining case coverage and reachability impossible since it cannot be ascertained whether a case subsumes another or can solve the problems that other similar cases can.

Another interesting pattern noticed in the related work on maintenance is that, barring Montaner *et al.* [MLdeR02], all other techniques implicitly assumed static problem domains, i.e. the data does not change with time. This is unrealistic in the software engineering field since it is common to see new technologies being used, which when gain popularity, make other cases irrelevant. But in the techniques discussed above, obsolescence and irrelevance seemed to be associated with non-use of the case for a threshold period of time. Hence, it is important for a framework to account for dynamic data to be implemented in software engineering.

One can also argue that it may be worth considering to discretize the solution which would turn it into a classification problem and enable applying the above mentioned techniques. But this may be unfavourable for numerous reasons. To cite a few, firstly industrial budgets require absolute values to optimise resource allocation. Secondly, the open-ended characteristic of the solution value causes selecting the range of intervals for discretisation challenging and acceptable to all. Thus, predicting within smaller ranges adds to uncertainity while using larger ranges is of lesser value when budgeting in the real world.

Thus, current techniques seem inadequate to be directly applicable to prediction problems for maintenance purposes. They either cannot be applied conceptually or lack measures to cater for continuous value solutions. Thus, having identified a gap in the research, this thesis aims to provide one possible alternative to tackle this problem. The subsequent chapters present the methodology for the proposed technique and then validate it on software engineering data sets.

II

Methodology

PAGE

MISSING

IN

ORIGINAL

CHAPTER 5

Meta-Data

This chapter provides a description of the method for generating the two building blocks of this research, which is termed as meta-data in this research. For ease of understanding, the methodology has been exemplified using one of the data sets from the analysis. To begin with, some instances of previous use of meta-data in CBR research are revisited. Thereafter, the proposed meta-data for this research are introduced followed by their generation procedure.

5.1 Meta-Data in CBR

The CBR community widely acknowledges that a within each case, there are features that describe the problem or represent a point in an n-dimensional problem space, while others (usually one) represent points in the solution space. Another perspective, as suggested by Finnie and Sun [FS02] is that the case base consists of a 'World of Problems' and a 'World of Solutions', i.e. $W_p \times W_s$. This is a more flexible outlook since it enables defining a $M : N$ relationships [EN03] between data points in the problem and solution spaces.

But, Leake and Wilson [LW98] and Reintartz *et al.* [RIRB01] advocated interpolating data sets with case-specific information that can be gainfully exploited during problem-solving. Though these suggestions were made from a maintenance perspective, similar data in case bases has been witnessed previously in research for a variety of purposes. Such data or features are typically unindexed that do not form a part of retrieval (i.e. are not used when computing inter-case similarity) but only provide background information about other data. While Reintartz *et al.* referred to such data as 'quality

information', in this research, it is referred to as 'meta-data' which has a more generic connotation.

Meta-data can be stored in three possible levels in the case base. The highest level would be the information stored that pertains to the case base itself. An example where such information can be vital is when using more than one case base [LS01, LS02a, LS02b]. On the next level, meta-data relevant to individual cases can be stored alongside the problem and solution space features. As mentioned earlier in Chapter 4, Portinale *et al.* [PTT99, PT00] stored case-specific meta-data including frequency of retrieval or reuse and frequency of successful and unsuccessful adaptations. Another example includes research by Kira and Arkin [KA04] who used features similar to Portinale *et al* for selecting cases to be deleted. On the lowest level, even individual features can possess their own meta-data such as their relevance or weighting [AB94] and descriptions that can be very useful when mapping features across heterogenous case bases.

The meta-data generated in this research is case-specific since the aim is to assess the regularity of the case base at that level and potentially identify noisy or unreliable cases. This chapter only describes the meta-data and the relevant methodology for their generation, while the following two chapters demonstrate how the meta-data can be used to accomplish the set objectives for this research.

5.2 Desharnais Data Set

Over the next three chapters (including this one), the methodology is exemplified using the Desharnais data set. The software engineering projects included in the data set originate from a Canadian software house. After removing 4 cases with missing solution values, the remaining 77 cases have been used in this research. Readers are directed to Section 8.2 (pp. 72) for descriptive statistics on the data set.

5.3 Distance and Residual Rank Ratios

In software engineering estimation, the problem and solution spaces are open-ended. As a result, while inter-case distances are normalised, the solution spaces are merely residuals* with no bounded ranges (i.e software engineering is an open-ended problem). For example, the range of the solution feature, *Effort*, in the Desharnais data set is 23394. This information says little about the quality of solution produced using nearest neighbour techniques, especially when this range is unknown to the user. Hence, the intention is to generate meta-data that is a further normalisation of (i.e. beyond normalising inter-case distance) problem and solution space distances, and are suitable for use in the techniques proposed in this research.

In [LW99], concepts of *PDist* and *RDist* were introduced where the former referred to problem space distances between two cases in a case base, while the latter referred to solution space distances. It was expected that these distances should exhibit at least some (preferably strong) regularity or association for a CBR system to be successfully functional. In this research, $DRR_{C,T}$ and $RRR_{C,T}$ are concrete instances of *PDist* and *RDist* respectively for prediction problem domains and are described in the following sub-sections.

Since the interest lies in assessing the quality of the case base before deployment, any proposed assessment must be carried upon only the constituent cases. This is accomplished by splitting the data set randomly and without replacement into a training set, Tr (which forms the case base) and the testing set, Ts (which comprises the test cases) in a $2:1$ ratio. This is a popular split ratio used widely in the machine learning community. The meta-data is generated only for the training set.

Of course, such a split may introduce a sample bias in the analysis and must be addressed. Hence, 30 such random samples of training and testing sets were generated for each data set used in this analysis. The number of samples (30) was motivated by an empirical investigation by Kirsopp and Shepperd [KS02b]. Their results highlighted that random split of data into a training and testing set may introduce sample bias causing the technique to produce untrustworthy results i.e. the technique's good or

*For example, the residual for the i^{th} observation of effort, e is given as $e_i - \hat{e}_i$, where e_i is the true value for effort and $\hat{e}_i$ is the predicted value.

poor performance may actually be an artifact of the sample. To overcome this shortcoming, they recommened testing the technique on at least 20 samples.

Thereafter, for each sample generated, two meta-data, Distance Rank Ratio and Residual Rank Ratio are generated. These are described in the following two subsections.

5.3.1 Distance Rank Ratio

The Distance Rank Ratio ($DRR_{C,T}$) is the ratio of the order of a candidate case's distance from the target case (with respect to the other candidate cases in the case base) to the other total number of cases in the case base less one (for which the retrieval is being made).

$$DRR_{C,T} = \frac{DistanceRank}{n-1} \tag{5.1}$$

This is given by Eqn. 5.1 which is the $DRR_{C,T}$ of candidate case C for target case T. *DistanceRank* is the candidate's position with respect to other candidate cases when sorted in increasing order of distance from the target. While n is the total number of cases in the case base. Hence, lower the value of $DRR_{C,T}$, the closer the candidate is to the target in a given case base. To exemplify, consider a case base with 51 cases. Now to predict for one of these cases, the Euclidean distance between it and the remaining 50 cases is calculated. The cases are then sorted by increasing order of their distance. Here, the closest case would have $DRR_{C,T}$ as $1/50 = 0.2$. The next similar case would have $DRR_{C,T}$ $2/50 = 0.4$ and so on until the furthest case with $DRR_{C,T}$ $50/50 = 1$.

5.3.2 Residual Rank Ratio

Likewise, Residual Rank Ratio ($RRR_{C,T}$) is the ratio of the order of a candidate case's residual from the target case (with respect to the other candidate cases in the case base) to the other total number of cases in the case base less one (for which the retrieval is being made). Recall, as explained earlier in the section, that residuals are the difference between the actual and predicted values.

$$RRR_{C,T} = \frac{ResidualRank}{n-1} \tag{5.2}$$

This is given by Eqn. 5.2 which is the $RRR_{C,T}$ of candidate case C for target case T. *ResidualRank* is the candidate's position with respect to other candidate cases when sorted in increasing order of residuals from the target. While n is the total number of cases in the case base. Here, lower the value of $RRR_{C,T}$, the better this candidate can predict for the target, assuming the solution is not adapted. Again, to exemplify consider a case base with 51 cases. To predict for one of these cases, the Euclidean distance between it and the remaining 50 cases is calculated. The cases are then sorted by increasing order of their absolute residual, i.e. in increasing order from the candidate case in the solution space. Here, the closest case would have $RRR_{C,T}$as $1/50 = 0.2$. The next similar case would have $RRR_{C,T}$ $2/50 = 0.4$ and so on until the furthest case with $RRR_{C,T}$ $50/50 = 1$.

To compute inter-case distances, Euclidean distance was used. Typically, software engineering data sets comprise both, numerical and categorical features. The distance metric, resultantly was modified slightly to accommodate both feature types. Given a case base with n number of software engineering projects (P) and $1 \leq i, j \leq n$, each described by a set of m features and $1 \leq l \leq m$; the distance between two projects, P_i and P_j is given by first, computing similarity between individual features (Eqn. 5.3) and then, summing them to compute total similarity (Eqn. 5.4).

$$\delta(P_{i,l}, P_{j,l}) = \begin{cases} \left(\frac{|P_{i,l} - P_{j,l}|}{max_l - min_l}\right)^2 & \text{if the } l^{th} \text{ feature is continuous,} \\ 0 & \text{if the } l^{th} \text{ feature is categorical and } P_{i,l} = P_{j,l}, \\ 1 & \text{if the } l^{th} \text{ feature is categorical and } P_{i,l} \neq P_{j,l}. \end{cases} \tag{5.3}$$

$$distance(P_i, P_j) = \sqrt{\frac{\sum_{l=1}^{m} \delta(P_{i,l}, P_{j,l})}{n}} \tag{5.4}$$

The relevant MATLAB code to compute Euclidean distance can be found in Appendices A.3 and A.3. It is important to note at this stage that missing data from the data

sets used in this analysis were removed since they require considerable research to be appropriately imputed or handled [TC05], or else may introduce bias.

5.4 Generation of Meta-Data

For each data set described in Chapter 8, 30 independent random samples of training and testing data were generated. The distance of each case (playing the target case) in the training set or case base, Tr from every other case (playing the candidate case) in the case base was computed along with the corresponding difference between their solution values, i.e. the residual. Thereafter, the values of $DRR_{C,T}$ and $RRR_{C,T}$ were computed as described in the previous section.

For each of the 30 random samples of the Desharnais data set, the training set comprised 51 cases while the testing set comprised of the remaining 26 cases. The meta-data was generated for each case in the case base as described above. Table 5.1 is an extract of the meta-data generated from the first of the 30 training sets. Column 1 denotes the case playing the role of the target for which a prediction is meant to be made. The second column denotes the case number against which the target case is compared to. In column 3, the Euclidean distance between the two cases is recorded while in column 4, the corresponding distance rank ratio is computed and recorded. Similarly, columns 5 and 6 record the corresponding residuals and residual rank ratios. Relevant MATLAB code is presented in the appendices on pages 136 and 137.

Hence, 2550 instances (i.e. 51 target cases × 50 candidate cases) of distances, residuals and corresponding $DRR_{C,T}$ and $RRR_{C,T}$ were generated. Table 5.1 exhibits the irregularities in the case base that we had hoped to perceive using the proposed meta-data. For example, `Case 14` is fairly similar to `Case 1` since $DRR_{C,T}$= 0.12 and as we would expect, its $RRR_{C,T}$ is very low too i.e. 0.02. Similarly, `Case 16` with very high $DRR_{C,T}$ also has a very high $RRR_{C,T}$. Such cases are examples of reliable cases since their behaviour is predictable, i.e. their problem and solution features are proportionally distant from the target's features. This is also in accordance with the consistence case property, as proposed in [RIRB00, IRRB04] and the CBR premise — '*Similar problems have similar solutions*'.

Table 5.1: Rank Ratio Example

Target	Candidate	Distance	$DRR_{C,T}$	Residual	$RRR_{C,T}$
1	2	0.3842	0.42	1498	0.56
1	3	0.4506	0.78	−973	0.36
1	4	0.5250	0.96	−301	0.14
1	5	0.3428	0.18	1127	0.42
1	6	0.3399	0.16	1316	0.48
1	7	0.1123	0.02	924	0.34
1	8	0.3839	0.40	1473	0.54
1	9	0.4832	0.88	455	0.22
1	10	0.3437	0.20	−7714	0.96
1	11	0.4762	0.86	483	0.24
1	12	0.4954	0.92	2247	0.68
1	13	0.3945	0.46	−1505	0.58
1	14	0.2215	0.12	21	0.02
1	15	0.2330	0.14	2030	0.62
1	16	0.5063	0.94	−5453	0.90

However, cases with unexpected patterns of behaviour need to be cautiously treated. Examples include `Case 10` having fairly low $DRR_{C,T}$, but would be a poor choice as a candidate case as reflected by its high $RRR_{C,T}$. Similarly, `Case 4` is very distant from the target case ($DRR_{C,T}$= 0.96) but makes an excellent candidate case given that $RRR_{C,T}$= 0.14. While this does not necessarily suggest that `Case 4` is unreliable, in this research, we consider it to be so. This is because reuse of `Case 4` when lacking sufficient implementation knowledge, which further results in the inability to adapt the solution, is likely to deliver a poor solution. Such cases are hence labelled unreliable cases since their problem and solution features are disproportionally distant from the target case.

5.5 Chapter Summary

Meta-data is potentially a valuable component in case bases that can be either generated or derived from other features to play crucial roles in the functioning of a CBR system. This chapter presented two novel meta-data that are the basic ingredients used in the techniques presented in this research, which are described in the following two chapters. A preview of the utility of the meta-data has been demonstrated on a sample of the Desharnais data. The meta-data helped observe the irregularity in the case base suggesting treating at least some cases with caution. This is because some cases behaved unreliably and their reuse may result in poor solutions. Of course, the data presented is only a small extract from a single sample, it is too early to draw any conclusions about the overall regularity of the Desharnais data set and its applicability to CBP.

But techniques to determine applicability of data sets to CBP by verifying the regularity in the case base is the focus in the next chapter. Thereafter, the focus drifts to techniques that assess individual case reliability that is further used to infer its suitability for reuse in an attempt to increase prediction accuracy.

CHAPTER 6

Case Base and Case Quality

Previously in Chapter 4, contemporary motivations and directions in case base maintenance research were cited that focussed upon the criticality of a good case base to deliver effective solutions. How the goodness or 'quality' of a case base is measured and the techniques implemented to retain such a case base is largely dependent on the domain, application and objective of the individual CBR system.

This chapter describes the techniques and methodology exploited in this research to gauge and visualise case base and individual case quality, or specifically regularity in this case, using the meta-data generated in Chapter 5 for CBP. Again, to ease understanding, the methodology is exemplified using the same Desharnais data set sample (first random sample) from the previous chapter. Firstly, techniques that measure case base quality are concentrated upon and then the focus moves to measuring individual case quality.

6.1 Mantel Randomisation Test

The Mantel Randomisation test* (Mantel's test) was primarily developed to compare two distance matrices (generated using a distance measure, e.g. Euclidean distance), and has so far been used across a range of disciplines such as ecology and biology [Man01]. Fundamentally, the test measures the association between corresponding elements of

*The author is indebted to Barbara Kitchenham [KKJ05] for her recommendation to use Mantel Randomisation test as one of the case base quality measures early on during the course of this research. In [KKJ05], the authors have used the Mantel test for feature subset selection to increase prediction accuracy.

two distance matrices using a suitable statistic (usually correlation). Assume that there are two distance matrices A and B :

$$A = \begin{bmatrix} 0 & a_{12} & \cdots & a_{1n} \\ a_{21} & 0 & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & 0 \end{bmatrix}, B = \begin{bmatrix} 0 & b_{12} & \cdots & b_{1n} \\ b_{21} & 0 & \cdots & b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \cdots & 0 \end{bmatrix}$$

Here, matrix A records the inter-case distances between the independent or predictor variables and hence can be referred to as predictor variables distance matrix, while matrix B records inter-case distances between the dependent or response variables and can be referred to as response variable distance matrix. In both matrices, elements a_{ij} and b_{ij} respectively record predictor and response variable distances between the i^{th} and j^{th} case. For example, in CBP matrix A will comprise distances amongst problem features set between cases in the case base while matrix B records the differences in solutions (residuals) between the respective cases. The diagonal elements in A and B are zero because in these cases, the candidate cases are same as the target (i.e. $i = j$).

$$R = \frac{\sum_{i,j=1}^{n} a_{ij}b_{ij} - \sum_{i,j=1}^{n} a_{ij} \sum_{i,j=1}^{n} b_{ij}/n}{\sqrt{\left[\left\{\sum_{i,j=1}^{n} a_{ij}^2 - (\sum_{i,j=1}^{n} a_{ij})^2/n\right\}\left\{\sum_{i,j=1}^{n} b_{ij}^2 - (\sum_{i,j=1}^{n} b_{ij})^2/n\right\}\right]}} \tag{6.1}$$

For the two given distance matrices A and B, Eqn. 6.1 calculates Mantel's test statistic (correlation in this case) where n is the square of the dimension of the matrices (both being square and of the same size) less the number of diagonal elements. The correlation (R_1) of the two distance matrices A and B is calculated by measuring across the pairs of corresponding elements excluding the diagonal elements. Thus, R_1 is an indicator of the degree of regularity in the case base. Higher values of R_1 would suggest corresponding data points are appropriately spaced out in the problem and solution spaces while lower values would indicate problem-solution irregularity in the case base.

Thereafter, the indices or positions of the elements of one of the matrices, say A, are randomised and the correlation (R_2) between original distance matrix B and the randomised distance matrix A is recalculated. Interestingly, only the term $\sum a_{ij}b_{ij}$ changes in Eqn. 6.1 when A is randomised and thus, is equivalent to R. Now, if $R_2 > R_1$, it would suggest that there exists no relationship between the predictor and response variables since random pairs are more strongly correlated. This is analogous to regression modelling where no independent variable has a beta coefficient significantly different from zero. Hence, to ascertain the likelihood of a relationship, Mantel's test statistic is calculated between matrix B and 4999 permutations of matrix A to test for statistical significance. The remainder of the section describes the implementation methodology.

It is important to recognise that Eqn. 6.1 is the formula to calculate correlation between pairs of any two given samples. The test's main contribution resides in the concept of measuring association between distance two matrices, one of whose elements' indices are rearranged randomly. Coupled with the test statistic, the randomisation ascertains the relationship between the predictor and response variables, computes its strength and tests for statistical significance or the likelihood of occurrence of the relationship by chance. Such a test is crucial to CBP systems since they function by assuming a strong relationship between predictor and response variables. Results would expose the degree of underlying irregularity in the case base and indicate if there is a need to incorporate additional pre-processing or functionality to enhance prediction accuracy or even disregard the case base from use.

One now arrives at the task of populating the two distance matrices for the problem domain. This task would seem as straightforward as recording inter-case distances (Euclidean distance in this case) for all predictor variables and recording them in matrix A and the corresponding residuals in the appropriate cells of matrix B. But, as pointed out earlier in Section 5.3, it is important for the distances to be appropriately normalised to be comparable to each other. Now, this is where the meta-data generated in Chapter 5 is first put to use.

To populate the two matrices, first, distance matrix A (predictor variables) is populated with values of $DRR_{C,T}$ (Section 5.3) using the corresponding target and candidate case numbers (Table 5.1) to serve as column and row indices respectively, e.g. $DRR_{C,T}$ for `target case 1` and `candidate case 2` is recorded in the second row and first

column of matrix A. Analogously, matrix B (response variables) is populated with the corresponding value of $RRR_{C,T}$ (Section 5.3) in the same cell position. This is continued sequentially for all generated instances of meta-data resulting in the two distance matrices A and B, which are now ready for Mantel's test. Recall that $DRR_{C,T}$ and $RRR_{C,T}$ are asymmetric (i.e. $DRR_{C,T} \neq DRR_{T,C}$ and $RRR_{C,T} \neq RRR_{T,C}$). Hence, using them for Mantel's test reflects the regularity of the case base from the perspective of providing good solutions to problems.

Once matrices A and B are populated, the correlation coefficient R_1 is calculated using Eqn. 6.1. Thereafter, another 4999 coefficients ($R_{2...5000}$) are calculated between $RRR_{C,T}$ and 4999 randomisations of $DRR_{C,T}$, this sample size of 5000 being adequate for reliable statistical testing [Man01]. The value of R_1 is examined to reflect upon the regularity of the case base and then is statistically tested using all 5000 correlation coefficients. The derived results are then analysed to judge the applicability of the case base to CBP and the degree of inherent regularity.

The relevant MATLAB code for this section can be found in the appendices on pages 139, 140 and Appendix A.6.

6.2 Visualising Case Base Quality

Though the Mantel's test is capable of measuring the problem-solution regularity in a case base, a visualisation that further substantiates the results can be very helpful. The following subsections present novel case base and case visualisation techniques that help examine case base regularity and the reliability of individual cases.

While the focus in this research is to visualise the regularity of the case base, previously there has been some interesting work on visualisation methods in CBR. To mention a few, these include CASCADE [MS01a] which enabled case base authoring by visualising the competence and coverage as cases are added to the case base. An extension to CASCADE was provided in [MS01b] by Mullins and Smyth. They presented a Spring Model, which basically is a force directed graph, that preserved the n-dimensional similarity relationships between cases in a case base and presented them on-screen. Such visualisations are very helpful to analyse the coverage of the domain by a case base since they reveal dense and sparsely covered areas. Such information

can be used to concentrate on collecting cases to cover the under-represented areas of the domain.

6.2.1 Overall Case Base Dissonance

First, a simple visualisation of case base regularity is provided that could confirm the results derived from the Mantel's test. Again, this is demonstrated using the Desharnais data set training sample from Chapter 5. Fig. 6.1 is a bubble plot of all 2550 pairs of $DRR_{C,T}$ and $RRR_{C,T}$ from Section 5.3 of candidate cases for each case as a target. The size of the bubbles increases in proportion to the frequency of data points superimposed, i.e. the frequency of pairs of $DRR_{C,T}$ and $RRR_{C,T}$ that share the same values. The grid-like distribution of data points in the figure is a consequence of rank normalisation, while their uniform spread across the entire plot highlights the dissonance inherent in the case base. A very regular case base would yield a high value of R_1 which further implies that retrieval instances with low $DRR_{C,T}$ would also yield low $RRR_{C,T}$ and vice versa. Hence, ideally the spread should be cigar shaped along the diagonal of Fig. 6.1 (within the enclosure across the diagonal in the figure) or even better, linear (i.e the data points lie on the diagonal itself), which would exhibit that cases with smaller distances would provide good solutions and distant cases would provide poorer solutions. Despite the observation in Fig. 6.1 that there is a higher concentration of larger bubbles in areas surrounding the diagonal, this sample exhibits considerable irregularity and indicates the necessity to be preprocessed to increase prediction accuracy.

Data points lying in the lower-left quadrant of Fig. 6.1 denote instances where candidate cases are close to the target case in both the problem and solution space e.g. `Case 14` in Table 5.1. Thus, given a case base with inherent inconsistencies, such candidate cases are usually reliable to be reused to deliver solutions. Similarly, data points on the upper-right quadrant denote instances where the target and candidate cases lie distant in both problem and solution space. A higher density of data points in these two quadrants indicates high problem-solution regularity of the case base. Candidate cases, which when used for prediction deliver $DRR_{C,T}$ and $RRR_{C,T}$ regularly pairing to lie in these two quadrants can be deemed reliable since they behave as one may expect given the rationale behind CBP.

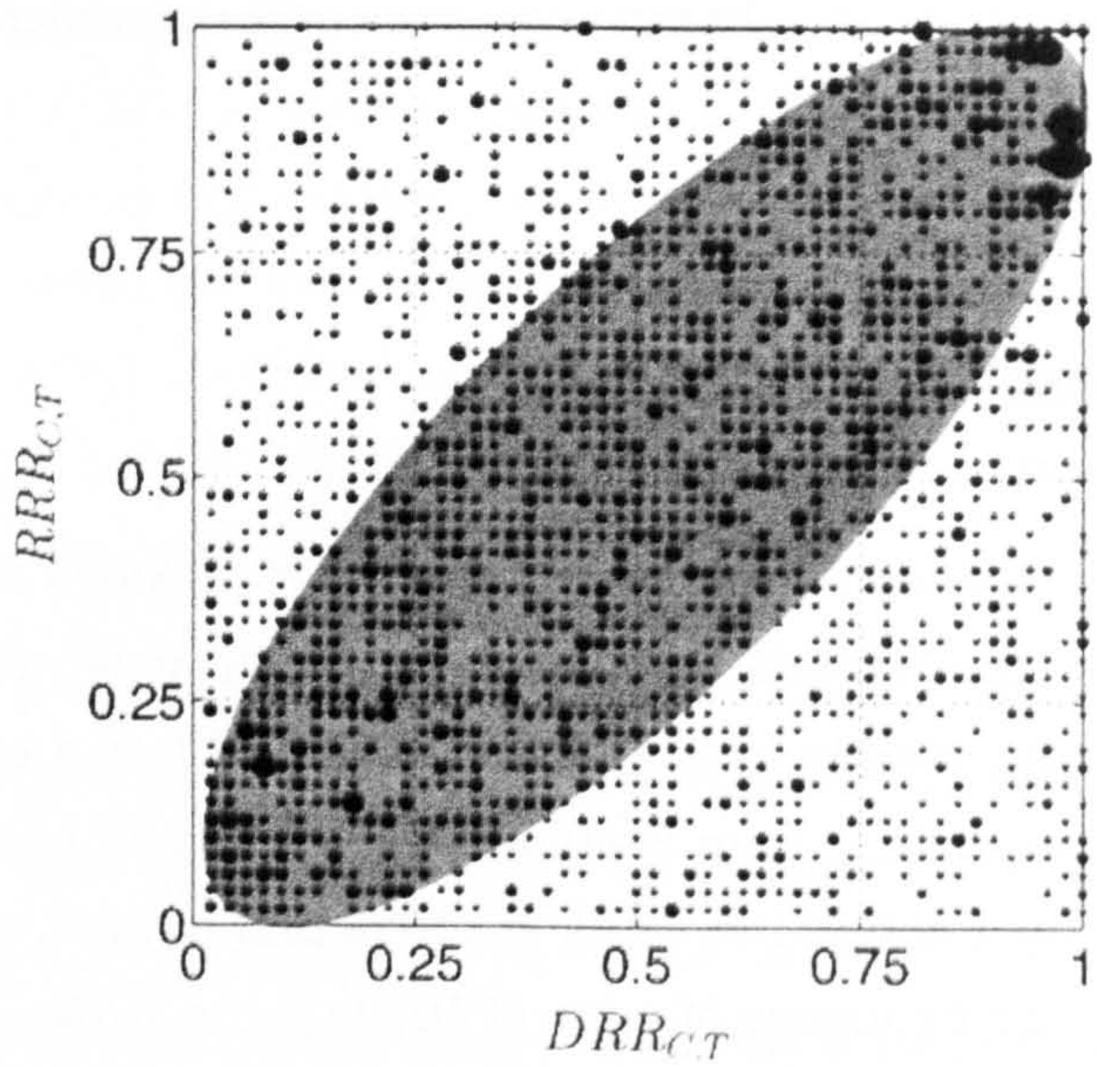

Figure 6.1: Distance Rank Ratio$_{C,T}$ Vs. Residual Rank Ratio$_{C,T}$

Whereas, retrieval instances represented by data points lying in the lower-right quadrant of Fig. 6.1 are those where the target and candidate cases are distant from each other in the problem space, but are remarkably close in the solution space, e.g. `Case 4` in Table 5.1. Analogously, data points lying in the upper-left quadrant of Fig. 6.1 represent those retrievals where the target and candidate cases are close to each other in the problem space, but are markedly distant in the solution space, e.g. `Case 10` in Table 5.1. The density of data points in the latter two quadrants signify the irregularity of the case base and suggests maintaining caution in its use.

6.2.2 Case Specific Dissonance

Fig. 6.1 reveals that the case base does exhibit inconsistency in the distribution of cases in the problem and solution spaces. However, what remains to be accomplished is the identification of such cases and a possible mechanism to reuse them cautiously. Hence, another plot is now introduced that reflects the same 2550 instances of meta-data, but from a different angle.

In Fig. 6.2, there are a series of rows and columns. Each column represents a target case corresponding to Column 1 in Table 5.1, while the rows (correctly interpreted from bottom-up) represent candidate cases sorted in increasing order of their $DRR_{C,T}$. Since

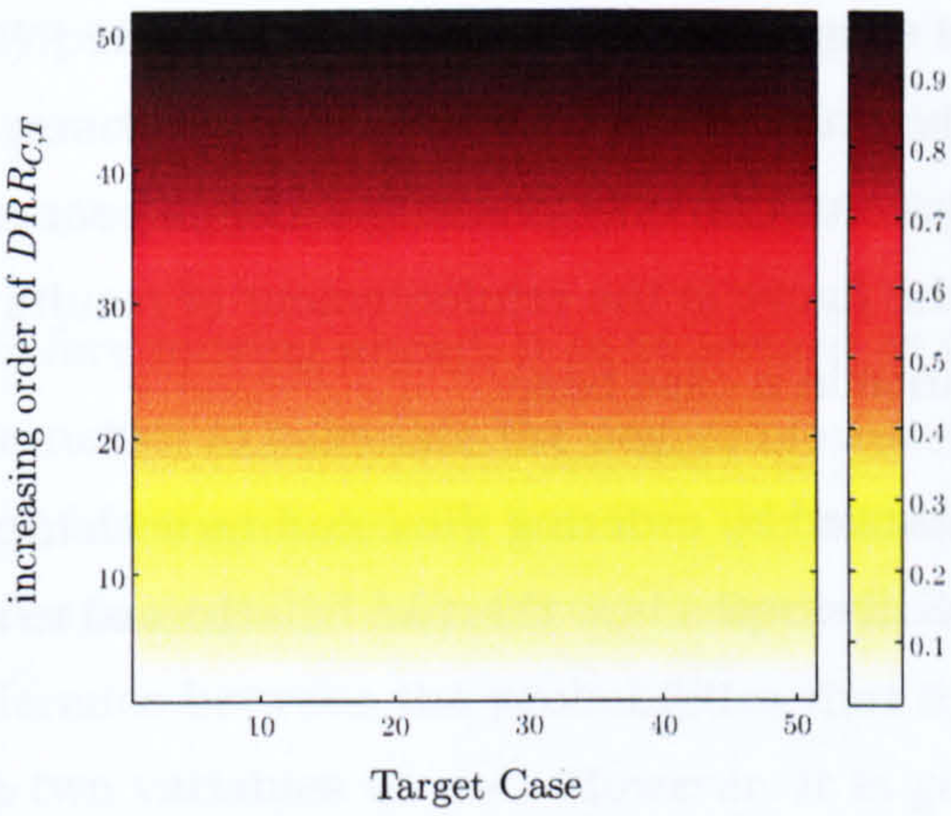

Figure 6.2: Sorted $DRR_{C,T}$ Matrix

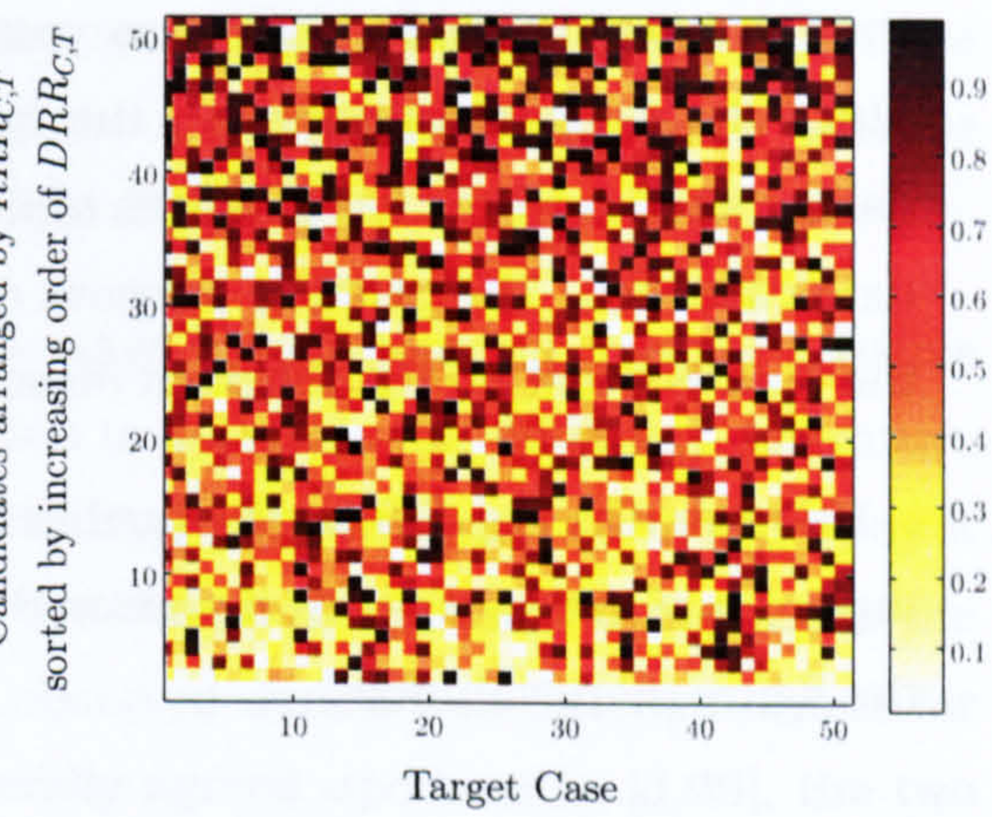

Figure 6.3: $RRR_{C,T}$ Matrix sorted by $DRR_{C,T}$

there are 51 cases in our case base, there are a total there are 51 rows and columns present in the figure. The different shades in the rows symbolise the distance rank ratio ($DRR_{C,T}$) of the candidate cases from the target. Paler blocks (i.e. the bottom-half of the figure) represent nearness to the target in the problem space. Conversely, darker blocks (i.e. the top-half of the figure) indicate that the candidate and target cases are very distant. The colour bar to the right of the plot indicates the shades associated with the corresponding range of values for $DRR_{C,T}$ ($[0-1]$).

Fig. 6.3 is of more importance in relation to Fig. 6.2. Once again, columns represent target cases, but rows represent $RRR_{C,T}$ of candidate cases that are arranged in the same order as in Fig. 6.2. The $RRR_{C,T}$ matrix of an ideal case base would have a similar distribution of colours as in Fig. 6.2. This is because one expects cases to be close to each other in both, the problem and solution spaces. However, from Fig. 6.3, it is clear that in reality, this may very well not be the case. Columns with paler colours in their lower half of Fig. 6.3 represent instances in the lower-left quadrant of Fig. 6.1 while darker colours represent instances in the upper-left quadrant. Conversely, paler blocks in the upper half of columns in Fig. 6.3 represent instances in the lower-right quadrant of Fig. 6.1 while darker colours represent instances in the upper-right. Again, the colour bar to the right of the plot indicates the shades associated with the corresponding range of values for $RRR_{C,T}$ ($[0-1]$).

Thus, Figs. 6.2 and 6.3 give a clearer picture of the case base and in addition, also help identify potentially unreliable cases that may be one of the root causes for the inherent inconsistency. The term 'potentially unreliable cases' has been used since

the Fig. 6.3 plots $RRR_{C,T}$ for every pair of target-candidate cases, and not the $RRR_{T,C}$. Since the ratio is asymmetric, the figure only suggests that the target cases, with darker shades in the lower halves of their columns, may possibly serve as poor candidate cases too. The primary purpose of the figure is the reinforcement of results from the Mantel's test on the inherent dissonance in a case base.

However, the Fig. 6.3 can be further improved by ordering the columns with increasing order of irregularity. Sub-section 6.2.3 discusses how this can be achieved to make the figure more valuable.

6.2.3 Spearman's Correlation

While Fig. 6.3 is a good indicator of inconsistency in a case base, it appears rather chaotic and makes it challenging to identify potential root causes (i.e. unreliable cases). Rearrangement of cases in Fig. 6.3 in decreasing order of their reliability would be a valuable transformation. Such a figure is likely to reveal the percentage of reliable cases in the case base and also, make identification of unreliable cases easier.

One alternative to achieve this is to measure the association of the series of $DRR_{C,T}$ and $RRR_{C,T}$ for every candidate case in the case base and then, sort cases by decreasing order of their association. For this, the Spearman's rank correlation* [FPPA91] is calculated between the corresponding pairs of $DRR_{C,T}$ and $RRR_{C,T}$. The formula for computing Spearman's coefficient (ρ) is given in Eqn. 6.2. Here, D is the difference in the ranks of $DRR_{C,T}$ and the corresponding $RRR_{C,T}$, and N is the number of pairs of $DRR_{C,T}$ and $RRR_{C,T}$.

$$\rho = 1 - \frac{6\sum_{i=1}^{N} D_i^2}{N(N^2-1)} \tag{6.2}$$

The Spearman's rank correlation measures the strength of association between two sets of data. When applied here, a high positive Spearman's correlation suggests that the candidate case has been very reliable in the past since its $DRR_{C,T}$ and $RRR_{C,T}$ are in accordance with each other. Conversely, a high negative value is a warning that the candidate has been very unreliable since its $DRR_{C,T}$ and $RRR_{C,T}$ from past instances of reuse move in opposite directions. The use of such cases in the future

*Spearman's rank correlation is preferred over Pearson's correlation for ordinal data.

may potentially continue to provide poor solutions. Lastly, a correlation value close to zero would indicate severe unpredictability about the case's performance because there is no clear pattern between its $DRR_{C,T}$ and $RRR_{C,T}$.

Before moving ahead, it is important at this stage to mention that another possible alternative to calculate the degree of association is Kendall's tau [FPPA91]. Literature suggests that Kendall's tau has the added advantage of being easy to interpret as it is fundamentally different from Spearman's correlation coefficient. It measures the difference between the probabilities that the observed data are in different orders for the two variables or not. However, it is generally agreed upon, as in [J.99], the two values are likely to be very similar to each other. Other than that, Spearman's rank correlation is still a popular choice and hence, has been adopted for this research.

Table 6.1: Case Quality using Spearman's Rank Correlation

Case No.	Correlation	Case No.	Correlation	Case No.	Correlation
1	0.33	18	0.60	35	−0.03
2	0.61	19	−0.19	36	0.45
3	0.25	20	0.20	37	0.25
4	0.59	21	0.19	38	0.66
5	0.43	22	0.33	39	0.37
6	0.30	23	0.38	40	0.30
7	0.25	24	0.36	41	0.33
8	0.70	25	0.68	42	0.06
9	0.46	26	0.34	43	−0.48
10	0.31	27	0.29	44	0.12
11	0.27	28	0.32	45	0.04
12	0.24	29	0.15	46	0.71
13	0.62	30	0.15	47	0.54
14	0.09	31	0.39	48	0.27
15	0.05	32	0.13	49	0.21
16	0.15	33	0.64	50	0.13
17	0.21	34	0.37	51	0.28

Table 6.1 shows the Spearman's Rank Correlation for each of the 51 cases in the

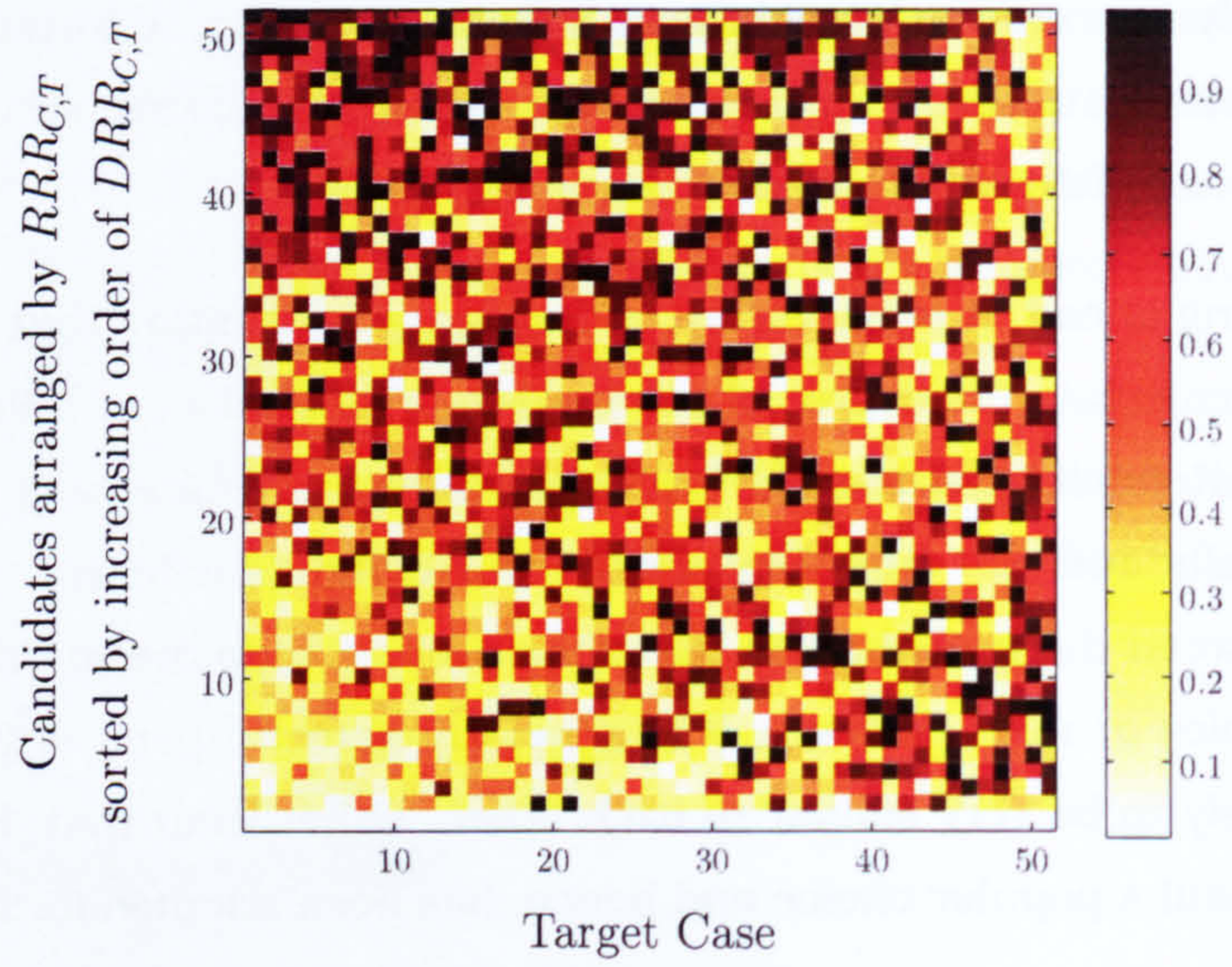

Figure 6.4: $RRR_{C,T}$ **Matrix sorted by** $DRR_{C,T}$

case base reflecting on the reliability of each case as a candidate. Here, `Case 46` has the highest correlation value of 0.71 suggesting that it is perhaps the most reliable case in our case base. This is further verified from column 46 in Fig. 6.3 where one can see a fair degree of concentration of paler colours in the lower half and darker colours in the upper half. On the other hand, `Case 35` is perhaps the most disorderly case in the case base having an absolute value of correlation closest to 0 at -0.03. Hence, cases of this nature may need to be used with extreme caution. This is because the prediction is nearly a guess when correlation ≈ 0. Lastly, `Case 43` has the highest negative correlation (-0.45) suggesting extreme caution before reuse to avoid a potentially poor solution.

Using the correlation values in Table 6.1, one can order columns in Fig. 6.3 in decreasing order of their reliability, i.e. their correlation values. Fig. 6.4 is a reordered alternative to Fig. 6.3 in which the correlation values have been used. It is easily deduced from the distribution of coloured blocks in the figure that case reliability decreases as one moves along cases in the case base. In this sample, nearly half the case base seems reliable (given the paler blocks in their column's lower-halves). The chaotic distribution of the coloured blocks for the remaining half suggests they are unreliable and must be used with caution.

Hence, visualisations such as Figs. 6.1 and 6.4 provide substantial insight into the

reliability of a case base. While the former figure reflected upon overall consistency, the latter figure indicated potential cases that can be deemed unreliable and as causes for overall case base inconsistency. However, the plots do not provide an objective measure of individual case reliability and it may be hard to examine very large case bases to single out potentially unreliable cases. Thus, in the following section, a new concept of case profiles are introduced that aid in measurement of case reliability.

6.3 Case Profile

Having established that the case base is irregular, it is crucial that cases significantly contributing to overall irregularity be identified and quarantined. Although the above visualisations aid in observing individual case reliability, their applicability is limited due to the subjectivity involved in assessing each case's performance as a candidate case and inefficiency associated with such examination for large case bases. Hence focus is now on gauging individual case reliability objectively to exploit such information during prediction in an attempt towards increasing accuracy.

The Spearman's Rank correlation gives a quick objective view of the case base quality. But the measure is still rather subjective and it is difficult to judge one case from another with similar correlation values. What is required is a mechanism that can distinguish one case as reliable or unreliable objectively. For this, case profiles are proposed to be generated for each case in the case base. A case profile is meant to track the performance of a case each time it has been reused and reflect its performance history as a candidate. This can be achieved by using the meta-data that has been used throughout this thesis in different ways. While this section concentrates on how to build case profiles, Section 7.2 explains how it can be used to discriminate against unreliable cases.

To build individual case profiles, the range of $DRR_{C,T}$ and $RRR_{C,T}$ ($[0-1]$) are divided into 4 equal intervals of size 0.25. This results in a matrix as Fig. 6.5. Scanning through the meta-data generated in Section 5.3 (2550 instances in this case), the count of each candidate case profile's cross-section quartile is incremented within which the $DRR_{C,T}$ and $RRR_{C,T}$ lie. For example, for any retrieval instance, if a candidate case

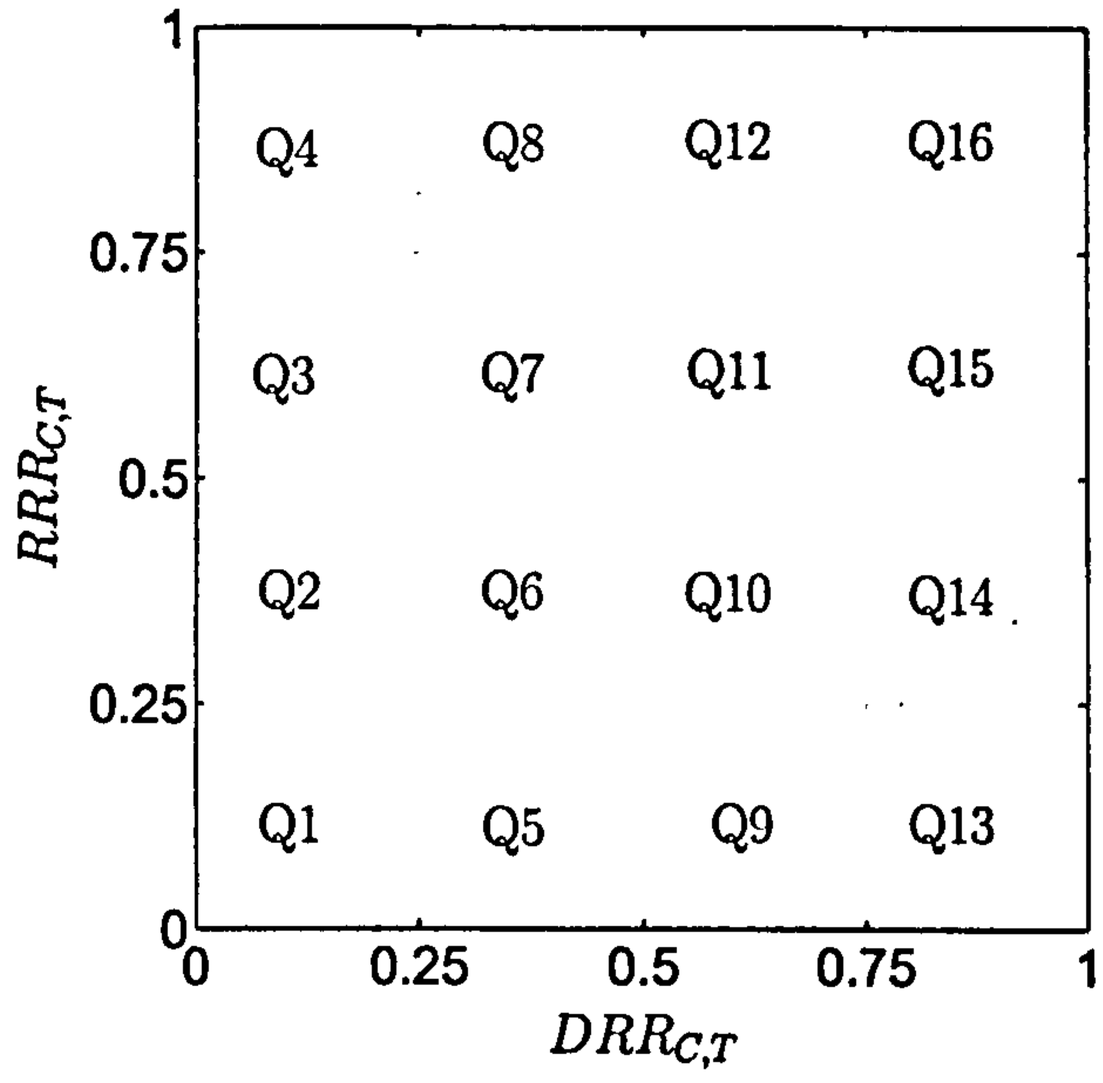

Figure 6.5: A Case Profile Example

has $DRR_{C,T}$= 0.125 and $RRR_{C,T}$= 0.3, $Q2$ was incremented by one or if $RRR_{C,T}$= 0.8, $Q4$ was incremented by one and so on.

It is important to note that the case profile is candidate case specific, unlike the case base visualisations plotted in Fig. 6.3 and 6.4. For a specific candidate case, its case profile is populated by taking into account the $DRR_{C,T}$ and $RRR_{C,T}$ for all instances of its reuse during the process of generating meta-data for the entire case base. Hence, the case profile is an apt reflection of the potential of a case to serve as a good candidate for reuse.

Fig. 6.5 can be superimposed on Fig. 6.3 and be interpreted likewise as in Section 6.2. A case with high density of data points in blocks $Q1$, $Q2$, $Q5$ and $Q6$ is desirable since its distance in the problem and solution space are proportional. Also, cases with higher density of data points in blocks $Q11$, $Q12$, $Q15$ and $Q16$ are equally desirable for the same characteristic as above. Hence, an ideal case may be one whose profile vastly covers the eight blocks discussed yet to form a form a cigar shaped distribution (see elliptical enclosure in Fig. 6.1).

While case profiles with data points lying in blocks $Q9$, $Q10$, $Q13$ and $Q14$ reflect large distances in the problem space but nearness in the solution space between target and candidate cases. These cases pose lesser risk since the likelihood of a good solution

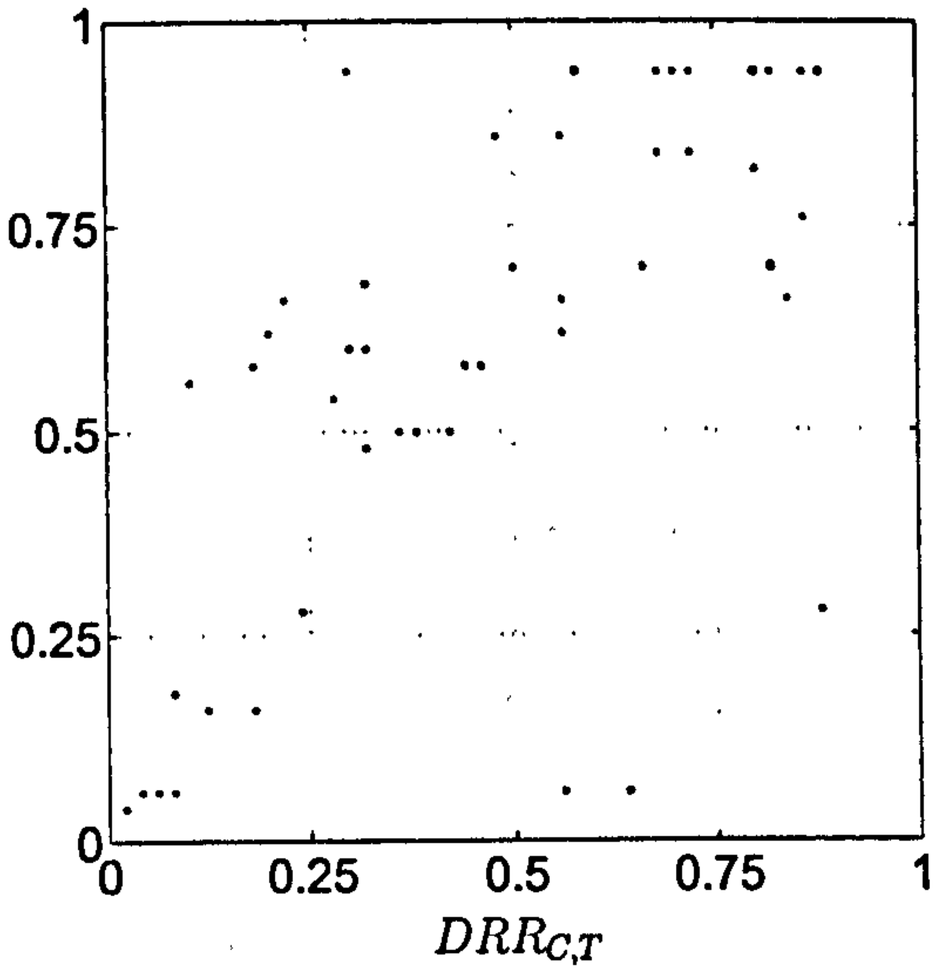

Figure 6.6: Reliable Case Example

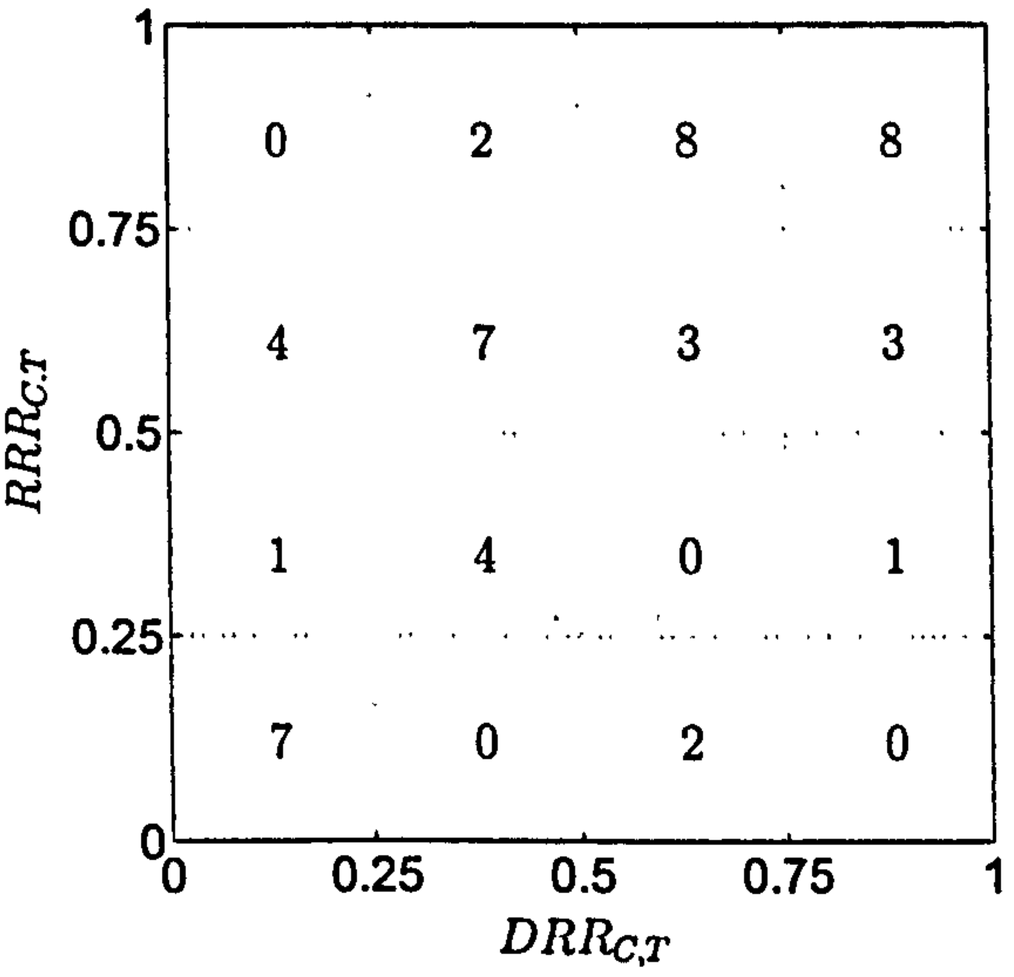

Figure 6.7: Reliable Case Example

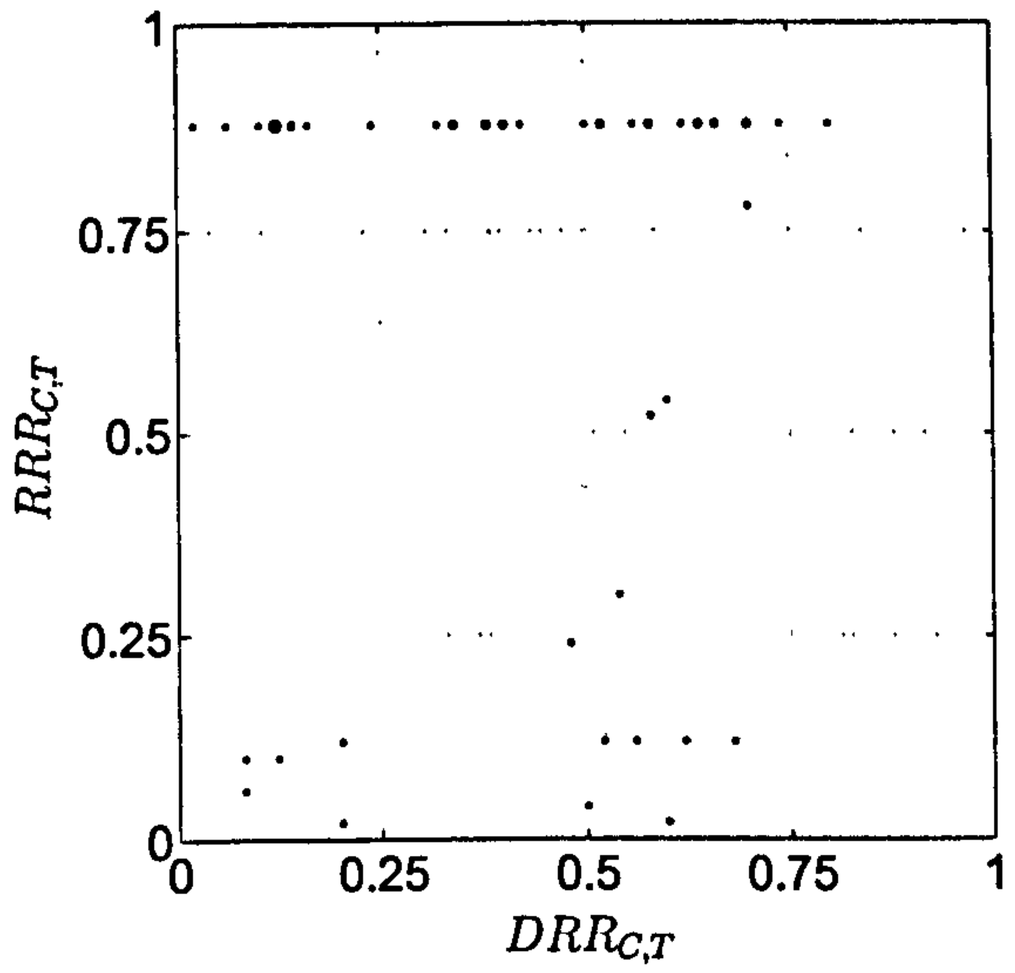

Figure 6.8: Unreliable Case Example

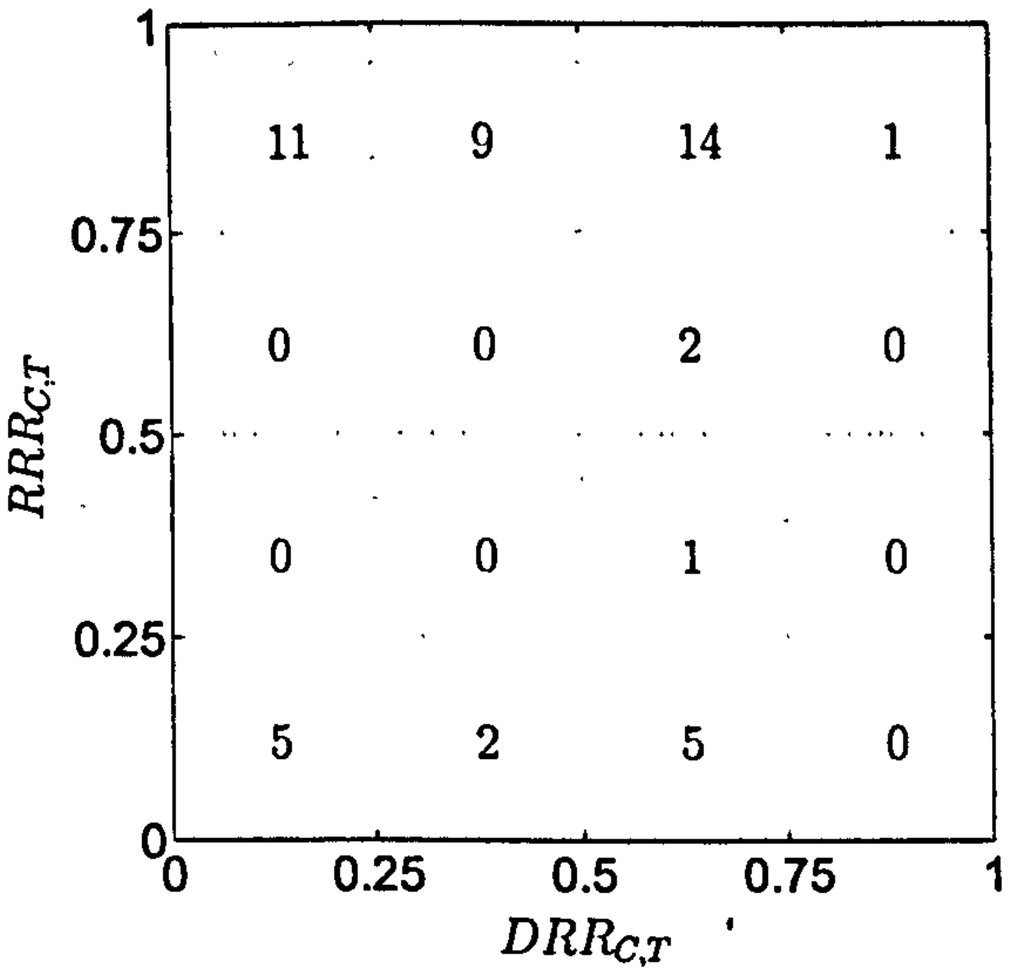

Figure 6.9: Unreliable Case Example

may be high, but this may be a result of unreliable cases balancing their performance. Conversely, case profiles with data points concentrated in blocks $Q3$, $Q4$, $Q7$ and $Q8$ signify nearness to the target case in the problem space but large distances in the solution space. Such cases are most important to be recognised due to the high probability of their reuse and delivery of a poor solution.

Figs. 6.6 and 6.8 demonstrate case profiles by visualising a reliable and unreliable case respectively. In the former figure, most data points can be found on the lower left and upper right quadrants of the profile. The few points in the lower left indicate

that in the problem space, this case is near few other cases and is at proportional proximities from them in the solution space too. While the higher concentration of points in the upper right quadrant signify that this particular case is largely dissimilar or distant to most other cases in the case base in both problem and solution spaces. Of course, there are a few instances in the upper left and lower right quadrants the denote instances of poor reuse. On the whole, the data points for this case are largely contained within the a cigar shaped distribution. The evidence of its reliability, which can be visualised, is that the few times this case would have been reused considering its distance from the target, it is likely to deliver a good solution. Fig. 6.7 supplements the case profile by reporting counts for the respective quadrants.

While in Fig. 6.8, one notices a rather more random distribution of data points strongly indicating the unreliability of the case in question. Although several data points are present in the lower left and upper right quadrants denoting regular behaviour, one also sees an ample number of data points distributed in the lower right quadrant. This indicates that although this case is far away from others in the case base in the problem space, it is relatively much closer to them in the solution space which is rather unexpected. It is noteworthy that there are also a few data points in the upper left quadrant which indicate that this case is also likely to deliver poor solutions when reused. Again, Fig. 6.9 supplements the case profile by reporting counts for the respective quadrants.

Relevant code for populating the case profiles can be found on pages 137 – 139.

6.4 Chapter Summary

This chapter described techniques employing the meta-data generated previously in Chapter 5 to gain insight into a case base. While the mantel's test delivers an objective measure of overall inherent regularity in the case base, the visualisation techniques are aimed at verifying and reinforcing its results. However, the visualisations provide a subjective impression of the quality of individual cases that are the root cause of overall distortion of the regularity of the case base. Hence, a new concept of case profile was introduced that tracked performance of a case from the perspective of it's capacity to be a candidate case that reliably delivers good solutions.

In the next chapter, a case discrimination system is presented that exploits such quality information recorded in the case profiles of individual cases to objectively determine their reliability. This attempt is in direction of attaining the larger goal of the research, i.e. to use reliable cases for prediction to enhance prediction quality.

CHAPTER 7

Enhancing Case-Based Prediction

With the larger objective of increasing solution accuracy, a case discrimination system is proposed that would work in tandem with the retrieval algorithm and reuse reliable cases. The proposed system is based on the idea of computing the likelihood of a candidate case to deliver an acceptable outcome. Failing to meet a threshold likelihood level, the CBR system would continue to seek the next nearest case that would satisfy the set performance criteria. The proposed technique is embedded within the *Retrieval* stage of the CBR cycle. During this stage, the CBR cycle retrieves select cases that qualify for reuse based on some objective metric such as distance from target, contextual relevance of case, adaptability [SK94]. For this analysis, the objective metric is the target–candidate case Euclidean distance. In addition, the technique endows the system with the candidate case's profile matrix and enables it to assess whether a candidate case can potentially generate a good solution. Once assessed, the relevant candidate case is chosen for reuse only if it meets a set performance threshold or else the next nearest case fulfilling the criteria is reused.

7.1 k-NN

For benchmarking purposes, the simple k-NN technique to is used compare and validate the technique's effectiveness. In CBP, k nearest neighbours of the target case are identified using a distance metric. The solution is then derived by statistically combining the solutions of each of the k cases. ArchANGEL [KS02a] provides several such methods for statistical combination (a form of solution adaptation) such as simple averaging, distance weighted average, rank weighted average and distance adjusted average. In this case, the nearest neighbours are identified by calculating the

euclidean distance between the target case and every candidate case in the case base. Then, the solutions of the nearest k neighbours are combined by computing their simple average to propose the final solution. Other alternative combining methods stated previously were not explored since this was neither the focus of this research nor any deep theory was found to motivate another choice.

Additionally, in previous research, Kadoda *et al.* [KCS00] experimented using values of k ranging from $[1,\ldots,5]$ and found $k = 3$ to generally provide the lowest residuals for the Desharnais data set. To maintain consistency, experiments are conducted adhering to the original choice of range for k.

7.2 Frequentist Approach for Case Assessment

Having available the meta-data for the case base, each constituent case bears an associated profile matrix conceptually in the form of Fig. 6.5. The matrix is populated with the frequency of good or poor solutions provided by a case when at a certain distance from the target case relative to all other cases in the case base. With access to this tabulated frequency matrix or the case profile, assessing the potential of a candidate case to deliver a good quality solution translates into determining the chance or likelihood of reusing the case and achieving a quality solution considering its distance from the target case. Thus, an obvious method of choice for such assessment is computing the probability of the event as:

$$Probability_{DRR_{C,T}} = \frac{Frequency of Good Solutions_{DRR_{C,T}}}{Frequency of Use_{DRR_{C,T}}}$$

This computes the probability ($[0 - 1]$) of a candidate case to deliver a good solution once its $DRR_{C,T}$ from the target case is known. But what remains to be answered in Eqn. 7.2 is what metrics should be used to define a good solution in order to calculate the respective frequencies.

In the current scenario, a good solution is identified by the $RRR_{C,T}$ of the candidate case. Since the candidate's solution is not modified (unless averaged for k nearest neighbours), cases with lowest $RRR_{C,T}$ are bound to deliver the best possible solution given the distribution of the solution space by the case base. Hence cases which, within

a given range of $DRR_{C,T}$, frequently provide solutions with low values of $RRR_{C,T}$ or have a higher concentration of data points in the lower halves of their profile matrices are to be preferred over others. Thus the likelihood or probability of a candidate case to provide a good solution is the ratio of the sum of its profile's lower blocks to the sum of all the blocks within the corresponding column whose range in which the $DRR_{C,T}$ lies. To exemplify, for a single case whose $DRR_{C,T}$= 0.2, probability is calculated as (Fig. 6.5):

$$P_1 = \frac{|Q1|}{|Q1| + |Q2| + |Q3| + |Q4|}$$

$$P_2 = \frac{|Q1| + |Q2|}{|Q1| + |Q2| + |Q3| + |Q4|}$$

Likewise, if $DRR_{C,T}$= 0.4, $|Q1|$, $|Q2|$, $|Q3|$ and $|Q4|$ would be replaced by $|Q5|$, $|Q6|$, $|Q7|$ and $|Q8|$ respectively and so on. Now, given a probability threshold limit PT, only those neighbouring cases whose values of P_1 or $P_2 > PT$ would be reused. In such a case, P_1 is relatively more discriminating than P_2 since it only considers the data points in the lowest quarter of the case profile as instances of providing good solutions to judge case reliance. Hence, a case may need to have performed exceptionally well in the past to meet a set threshold. On the other hand, P_2 is more permissive since it considers all data points in the lower half of the case profile to compute the probability or case reliability. Hence, the chances of cases being accepted for reuse increase using the latter equation given the typical size of software engineering data sets. This however remains a question to be examined.

Another parameter that can vary the intensity of case discrimination for reuse is the value of PT. The value of PT can lie between [0 – 1] where setting $PT = 0$ is equivalent to using k nearest cases (since there is no discrimination) while $PT = 1$ would expect all data points for a case to lie in the lowest quarter or lower half of the profile matrix. Hence, an in-between value is required so that the system would neither be too permissive nor too discriminatory .

In this analysis, the range ([0–0.8]) for PT has been experimented with. The range begins begins with 0 since it is equivalent to using the k nearest neighbours against which the rest of the solutions will be benchmarked. However, during the coding stage,

it was discovered that setting $PT > 0.8$ caused prediction to fail since the system was unable to find any cases that meet such stringent criteria. Portinale too claimed that setting quality levels very high would case the system to fail at finding suitable cases to deliver a solution [PT00].

7.3 Procedure

This section describes the procedure followed to conduct the experiments. There are three parameters (k, P_1, P_2 and PT) whose individual and combinatorial values would affect case selection and consequently prediction accuracy. A range of possible combinations of k and PT were experimented with for every random sample of each data set.

For a test case TS_i, whose solution is to be predicted, its Euclidean distance from every candidate case is computed. Then, the k nearest cases are averaged to provide a solution. Thereafter, the first k nearest neighbours for which $P_1 > PT$ are identified to provide a solution and lastly, the first k nearest cases for which $P_2 > PT$ are identified to provide a solution. This is repeated for each test case covering every combinatorial values of k, P and PT and the sum of absolute residuals for each combination is stored for comparing accuracy later. The relevant MATLAB code has been presented in the appendices on pages 140 – 142, Appendices A.7 and A.8.

7.4 Analysis of Results

Sum of absolute residuals ($Sum|Res|$) was chosen to measure error since it has largely been accepted to be an unbiased statistic [KPMS01]. Also, it is assumed that the direction of error is immaterial as both underestimation and overestimation must be precluded. Of course, this may not necessarily true for all software companies and more so in other domains where one may favour underestimation to overestimation or vice versa. In such situations, an appropriate measure of error must instead be used. For each random sample of the data set, 90 $Sum|Res|$ resulting were derived from 90 possible combinations of k, P_1, P_2 and PT (i.e. $5 \times 2 \times 9$ respectively). Also, these 90 $Sum|Res|$ will be derived for each of the 30 random samples of every data set.

The results were analysed by reporting the means of the $Sum|Res|$ for each combinatorial value of the three paraments from all 30 random samples. However, since the residuals are not normally distributed, a Kruskal-Wallis [Kan99] test was performed on the residuals using only k and the combination of k and PT that gave us the lowest mean of sum of absolute errors. The Kruskal-Wallis test is a non-parametric form (comparing medians) of ANOVA that determines if the samples have come from different populations. Hence the null hypothesis was:

> H_O: there are no differences between the medians of the samples i.e. using PT in conjunction with k-NN makes no difference to prediction accuracy.

The alternate hypothesis is:

> H_A: there are differences between the medians of the samples i.e. using PT in conjunction with k-NN makes a difference to prediction accuracy.

7.5 Chapter Summary

This chapter has presented the proposed technique that uses individual case profiles to judge the reliability of a case to serve as a candidate. The reliability of the case is gauged by simply computing the probability of the case to deliver a good solution. Then the case is reused only if it meets a set threshold of reliability, else the next nearest case with an adequate reliability is reused.

It is expected that reuse of such reliable cases would help increase prediction accuracy. The remaining parts of the thesis are directed at investigating the value of the proposed technique. The next chapter introduces the data sets used in the analysis and includes a brief statistical summary of each. And the following chapter includes the results from administering all the techniques proposed in Chapters 5, 6 and 7.

III

Results and Discussion

PAGE

MISSING

IN

ORIGINAL

CHAPTER 8

Data Sets

In this chapter, the three software engineering data sets, *viz.* British Telecom (BT), Desharnais and Finnish, are presented which were used to test the proposed techniques. Their selection was largely governed by their respective sizes to enable examination of the effect or impact of using the techniques on case bases of different sizes, and the required computational expense. Another motivating factor was the research group's familiarity with these data sets since they have been previously used in various publications.

In the following sections, for each data set, a description has been provided including a brief statistical summary of their respective features including the mean, median, range and standard deviation. This is to give the reader an idea of the expanse of the problem and solution spaces commonly dealt with in the this problem domain.

8.1 British Telecom Data Set

The BT data set comprising 18 cases is the smallest data set analysed in this research. The features are presented in Table 8.1. It was collected in the mid-90s for the purpose of estimating testing costs. The projects involve an approximate total of 4 million lines of code in C.

Five of the total eight features in the data set were included to compute inter-case Euclidean distances. The remaining three were removed for various reasons including being redundant or case labels.

Table 8.1: BT Data Set Feature List

Feature	Description	Range	Median	Mean	Std. Dev.
Case Name	Case Label	N/A	N/A	N/A	N/A
EstTotal	Estimated no. of hours	30 – 777	289	320.3	240.1
ActTotal	Actual no. of hours	24 – 1116	222	284.28	264.85
ActDevEffort	Actual devp. effort	16 – 833	153.5	202.61	193.2
ActTestEffort	Actual test. effort	7 – 267	50.5	80.83	75.38
Changes	No. of changes	3 – 377	127	138.05	120
Files	No. of files	3 – 284	68.5	100	92
Development Type	N/E Interfaces Architectural Business	N/A	N/A	N/A	N/A

8.2 Desharnais Data Set

A medium sized data set comprising 77 complete and 4 incomplete software projects, the Desharnais data set was collected by Jean-Marc Desharnais [Des89] from a Canadian software house. These projects were collected across three different developmentenvironments. All but one of the 11 features are numeric while the remaining feature 'Language' is categorical and denotes 3 development environments. Of the 77 completed projects, 44 (57%) belonged to development environment 1, while the remaining 23 (30%) and 10 (13%) projects belonged to environments 2 and 3 respectively.

8.3 Finnish Data Set

This data set is derived from the 2004 release of the "Experience data set" usually referred to as the Finnish data set and some of its features have been analysed in depth by our research group [PSKF05]. The Finnish data set is a result of a commercial initiative by Software Technology Transfer Finland (STTF) to provide support for software development organisations for both project cost estimation and productivity analyses.

Table 8.2: Desharnais Data Set Feature List

Feature	Description	Range	Median	Mean	Std. Dev.
Effort	Total project effort measured in hours	546 – 23940	3542	4834	4188
ExpEquip	Team experience in years	0 – 4	2	2.3	1.33
ExpProjMan	Project managers experience in years	0 – 7	3	2.65	1.52
Length		1 – 36	10	11.3	6.8
Trans	Number of transactions	9 – 886	134	177.5	146
Entities	Number of entities	7 – 387	96	120.6	86.12
RawFP	Unadjusted function points	62 – 1116	247	282.4	186.4
AdjFP	Adjusted function points	73 – 1127	258	298	182.3
AdjFact	Adjustment factor	5 – 52	28	27.4	10.5
YearFin	Year of completion	1983 – 1988	N/A	N/A	N/A

This has resulted in a data set which includes software projects between 1978 and 2003 and in its current form, it comprises 622 projects. Organisations pay an annual fee to gain access to the data via a tool called Experience Pro. The same tool is used to submit their own project data upon which they are entitled to a discount on their annual fee. The use of the tool for project data submission facilitates standardisation of variables included. In addition, the project data are carefully assessed at STTF by experts before being added to the data base. More information about Experience Pro is available at their website [Exp].

The projects are derived from a wide range of business sectors spanning financial to telecommunication projects and embrace a range of different platforms and development technologies. The data set includes both New Development (approximately 93% of observations) and Maintenance projects. Project data include size information in function points (FPs), effort and a range of factors to characterise the type of project,

factors to characterise the development circumstances, development and target technology. In total, 102 variables are collected. A fuller description of a previous version of the data set may be found in [MF00].

The major hurdle faced in using this data set was the presence of several missing values. Hence, it had to be substantially reduced in size for actual use in this analysis. In addition, several redundant and irrelevant features were also removed. Table 8.3 includes all remaining features after cleaning the data set and were included in this analysis. Here, Effort was the dependent variable while the remaining features were used to measure inter-case similarity. A total of 137 complete projects projects remained after cleaning and were used for the analysis. Symbols have been placed after some feature names in Table 8.3 indicating the feature type and hence the corresponding summary statistics. Features with no symbols are continuous value features whose mean, median, range and standard deviation have been reported. Features with symbols † are categories and no statistics are reported for these figures. Lastly, features with symbols ⋆ are ordinal features and their mode has been reported along with their range. The mean and standard deviations for these features have been excluded since they are ordinal values and are in effect, categories. Irrelevant columns for the respective features have been labelled N/A.

8.4 Chapter Summary

The three software engineering data sets used in this research have been introduced in this chapter along with a brief descriptive statistical summary for each. The main idea behind their choice was to examine the techniques performance on data sets of different characteristics, especially size and inherent regularity. The next chapter presents the results from applying the techniques on the data sets which is followed by a discussion on their effectiveness on each of them.

Table 8.3: Finnish Data Set Feature List

Feature	Description	Range	Median	Mean/Mode	Std. Dev.
Effort	Duration of project in hours	55 – 21994	2885	1655	3486
SizeUnit	Function Point measure (for all projects EP20)	N/A	N/A	N/A	N/A
StartDate	Year of commencement of project	1997 – 2003	N/A	N/A	N/A
HardWare†	Target platform	N/A	N/A	N/A	N/A
Business†	Business Sector	N/A	N/A	N/A	N/A
DevLang†	Programming Language	N/A	N/A	N/A	N/A
YK†	Company code	N/A	N/A	N/A	N/A
AmountofApps	No. of applications developed	1 – 14	2	1	1.7
ProjSize	Size in EP20	27 – 5060	498	329	592
Reusemultiplier		0.64 – 1	0.98	1	0.044
SituationAnalysisModel†		N/A	N/A	N/A	N/A
SituationAnalysisMultiplier		0.2 – 2.48	1.1	1.02	0.41
CompDeliveryRate		0.5 – 31.8	6	5.26	4.2
T01⋆	Involvement of customer representatives	1 – 5	N/A	3	N/A
T02⋆	Performance and availability of development environment	1 – 5	N/A	3	N/A
T03⋆	Availability of IT staff	1 – 5	N/A	3	N/A
T04⋆	No. of stakeholders	1 – 5	N/A	3	N/A
T05⋆	Pressure on schedule	1 – 5	N/A	3	N/A
T06⋆	Impact of standards	1 – 5	N/A	3	N/A
T07⋆	Impact of methods	1 – 5	N/A	3	N/A
T08⋆	Impact of tools	1 – 5	N/A	3	N/A
T09⋆	Level of change management	1 – 5	N/A	3	N/A
T10⋆	Maturity of software development process	1 – 5	N/A	3	N/A
T11⋆	Logical complexity of software	1 – 5	N/A	3	N/A
T12⋆	Size of database based on number of entities	1 – 5	N/A	3	N/A
T13⋆	No. of interfaces to other software	1 – 5	N/A	3	N/A
T14⋆	Quality requirements of software	1 – 5	N/A	3	N/A
T15⋆	Efficiency requirements of software	1 – 5	N/A	3	N/A
T16⋆	Training and installation/platform requirements	1 – 5	N/A	3	N/A
T17⋆	Analysis skills of staff	1 – 5	N/A	3	N/A
T18⋆	Application knowledge of staff	1 – 5	N/A	3	N/A
T19⋆	Tool skills of staff	1 – 5	N/A	3	N/A
T20⋆	Experience of project mgmt.	1 – 5	N/A	3	N/A
T21⋆	Team skills of project team	1 – 5	N/A	3	N/A
WBSModel†	Work Breakdown Structure	N/A	N/A	N/A	N/A

CHAPTER 9

Results

This chapter presents and discusses results from implementing our proposed techniques on the data sets described in the previous chapter. For each data set, the analysis begins with Mantel's randomisation test and then moves on to addressing the regularity of the case bases visually and objectively. Lastly, we demonstrate the use and impact of generated case-specific knowledge to enhance prediction accuracy.

9.1 BT Data Set

The BT data set comprising 18 cases was split into 30 random samples of training and testing sets as described earlier in Section 5.3. Thus, in each sample the training set comprised 12 cases and the remaining 6 cases constituted the testing set. Additionally, 132 instances (i.e. $12 * 11$) of meta-data were generated for each sample (Chapter 5).

9.1.1 Mantel Randomisation Test

The results from performing the Mantel test (concept and methodology discussed in Section 6.1) on the data set are first examined to gauge overall inherent problem-solution regularity. These, from each of the 30 samples, have been recorded in Table 9.1 including the correlation between the original distance matrices and the maximum and minimum values from the 4999 correlation coefficients between the §random and original matrices.

Several interesting observations can be made from Table 9.1. Firstly, all 30 coefficients between the original distance matrices are positive. This indicates that from

a hypothetical reference point, corresponding data points in the problem and solution spaces tend to move in the same direction. Such a characteristic of the data is particularly important for models based on empirical data since they rely upon a positive (or negative depending upon the objectives) association between the predictor and response variables.

Secondly, not only are the original coefficients positive, they are also distributed over a wide range of values. The highest recorded correlation was 0.70 for `sample 6` and the lowest being 0.30 for `sample 10`. Such high variation is an artifact of the small size of the training set (Tr) and the likely presence of some unreliable cases that distort the regularity of case bases with low coefficients. Also, previous research by Kirsopp and Shepperd addressed issues when dealing with small data sets that is likely to include sample bias [KS02b]. Case bases with high correlation coefficients are likely to attract more user confidence since they reflect inherent consistency in the distribution of points in the problem and solution spaces. While using CBP systems endowed with such case bases, a user could expect a near optimal solution (i.e. the best the case base can provide) to the problem, i.e. a solution with low $RRR_{C,T}$, without the aid of additional case or domain knowledge. On the other hand, systems using case bases with low coefficient values may need to be augmented with more knowledge of individual cases to afford a reflective choice of cases for reuse.

Lastly, each random sample's association between the predictor and response variables was statistically significant at $p < 0.001$ or at 0.1% level. This strengthens the likelihood of a true relationship between the two sets of variables in the real world. It is noteworthy that in 29 out of the 30 samples analysed, the maximum recorded correlations were equal to that between the original matrices in 20 cases. These results support the use of the BT data set for CBP since a system strictly using the k-NN technique for reuse would benefit from using the original combinations of predictor and response variables rather than any random association (which would make it hard to explain causality). However, in the remaining sample, 1 combination was more strongly associated with each other in comparison to the original pairs of variables. The low correlation coefficient of this sample indicates the presence of unreliable cases. Still, the sample remains statistically significant, but is very unlikely to distort solution accuracy.

Thus, the Mantel test results confirm that the BT data set does contain at least few

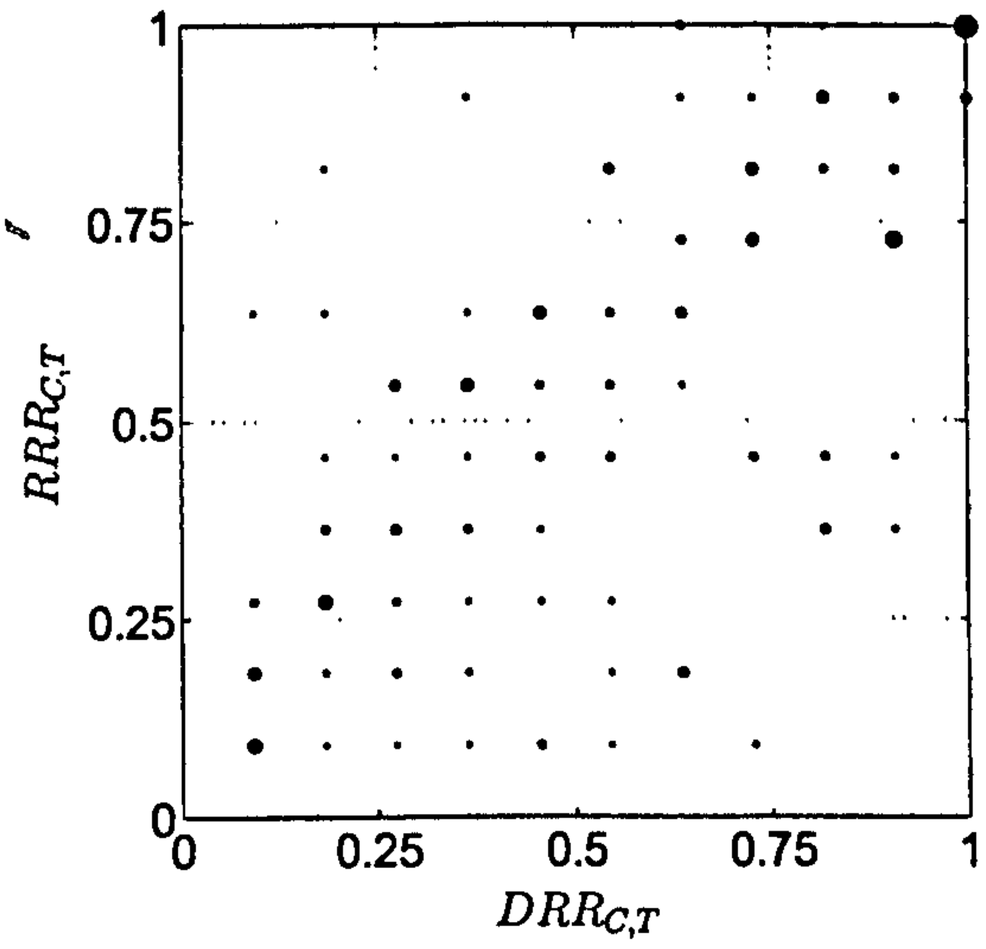

Figure 9.1: $DRR_{C,T}$ vs. $RRR_{C,T}$ for Regular BT Sample 6

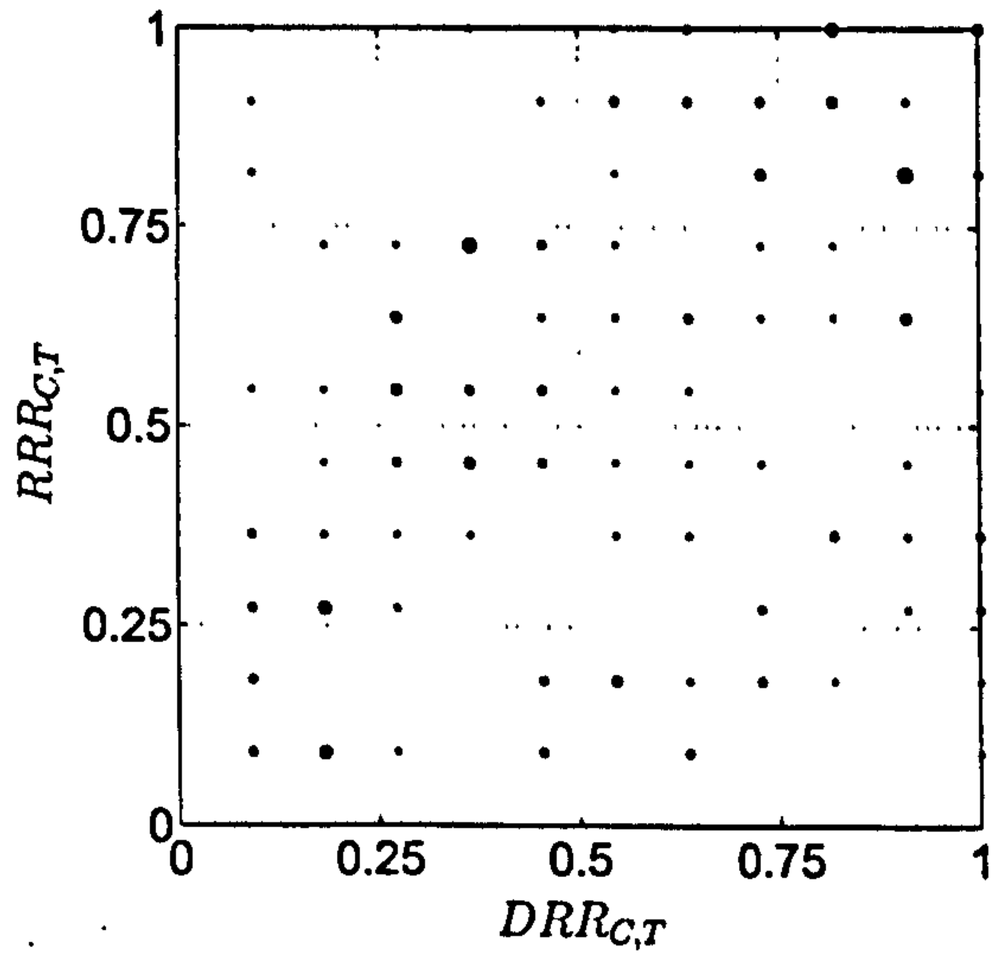

Figure 9.2: $DRR_{C,T}$ vs. $RRR_{C,T}$ for Irregular BT Sample 10

unreliable cases that distort the overall regularity of the case base. This is evident from the wide range of correlation coefficients of the samples. Also, each coefficient is positive and many being on the higher end warrant the use of the data set for CBP. Although each sample's coefficient was statistically significant, the tests signal the importance of some pre-processing of the case base before deployment to ensure relatively more dependable solutions are proposed.

9.1.2 Visualisation Case Base Quality

Having identified the most regular and irregular random samples from the mantel test, now the utility of the proposed visualisation techniques is demonstrates to verify the test's results. Figs. 9.1 and 9.2 are scatter plots of $DRR_{C,T}$ and $RRR_{C,T}$ from random samples 6 and 10 respectively. Recall that these samples have correlation coefficient values of 0.70 and 0.30 between predictor and response distance matrices.

The spread of data points in Fig. 9.1 is more close to cigar-shaped (Fig. 6.1) and represents a relatively consistent case base. The larger concentration of points in the lower-left quadrant signify retrieval instances with low $DRR_{C,T}$ and low $RRR_{C,T}$, implying closeness of target and candidate cases in both problem and solution spaces. Similarly, the high density of data points in the upper-right quadrant denotes instances where target and candidate cases were far apart in the problem and solution

Table 9.1: Mantel's Randomisation Test Results on the BT Data Set

Sample		Corr.	Sample		Corr.	Sample		Corr.
1	Max.	0.43	11	Max.	0.31	21	Max.	0.47
	Original	0.43		Original	0.31		Original	0.47
	Min	-0.31		Min	-0.27		Min	-0.29
2	Max.	0.35	12	Max.	0.68	22	Max.	0.45
	Original	0.35		Original	0.68		Original	0.45
	Min	-0.28		Min	-0.23		Min	-0.23
3	Max.	0.67	13	Max.	0.43	23	Max.	0.38
	Original	0.67		Original	0.43		Original	0.38
	Min	-0.24		Min	-0.28		Min	-0.27
4	Max.	0.58	14	Max.	0.57	24	Max.	0.32
	Original	0.58		Original	0.57		Original	0.32
	Min	-0.26		Min	-0.29		Min	-0.28
5	Max.	0.66	15	Max.	0.47	25	Max.	0.69
	Original	0.66		Original	0.47		Original	0.69
	Min	-0.31		Min	-0.35		Min	-0.33
6	Max.	0.70	16	Max.	0.57	26	Max.	0.33
	Original	0.70		Original	0.57		Original	0.33
	Min	-0.28		Min	-0.26		Min	-0.26
7	Max.	0.58	17	Max.	0.45	27	Max.	0.40
	Original	0.58		Original	0.45		Original	0.40
	Min	-0.28		Min	-0.26		Min	-0.30
8	Max.	0.39	18	Max.	0.31	28	Max.	0.40
	Original	0.39		Original	0.31		Original	0.40
	Min	-0.28		Min	-0.30		Min	-0.28
9	Max.	0.42	19	Max.	0.40	29	Max.	0.35
	Original	0.42		Original	0.40		Original	0.35
	Min	-0.23		Min	-0.25		Min	-0.26
10	Max.	0.36	20	Max.	0.43	30	Max.	0.44
	Original	0.30		Original	0.43		Original	0.44
	Min	-0.30		Min	-0.27		Min	-0.23

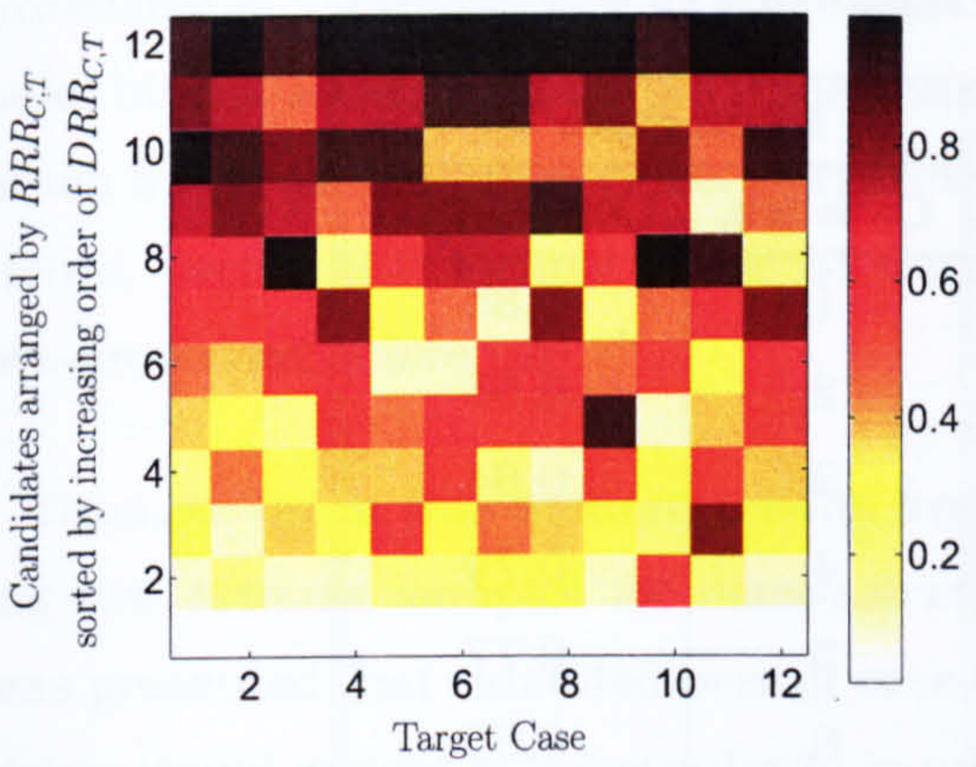

Figure 9.3: $RRR_{C,T}$ sorted by $DRR_{C,T}$ for Regular BT Sample 6

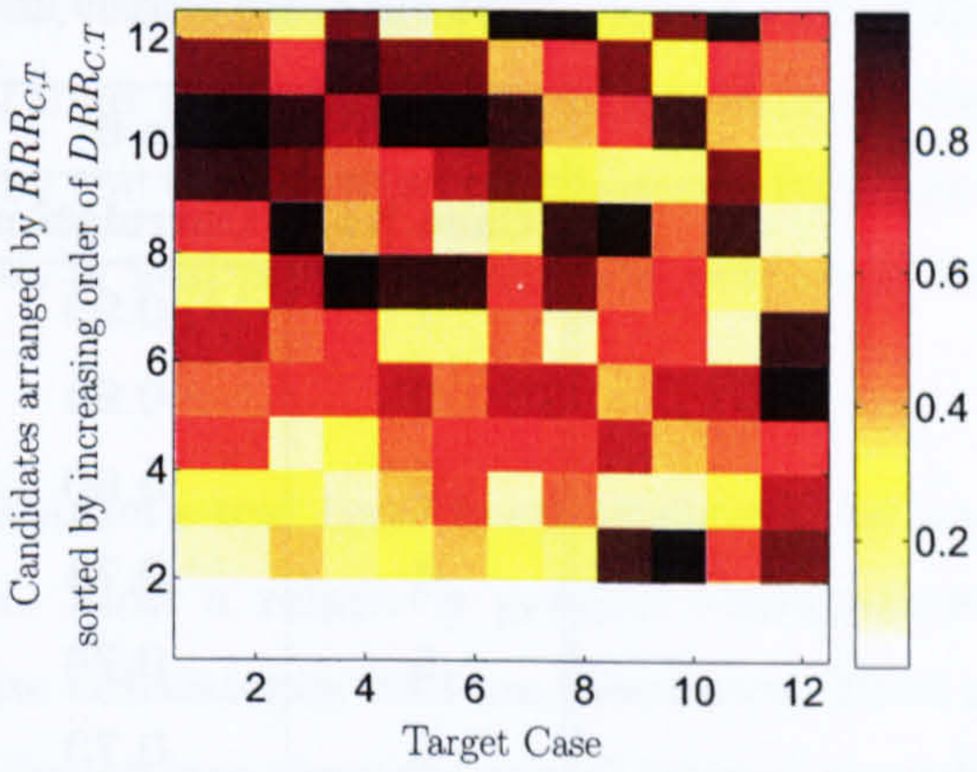

Figure 9.4: $RRR_{C,T}$ sorted by $DRR_{C,T}$ for Irregular BT Sample 10

spaces. Additionally, very few data points lying in the upper-left and lower-right quadrants augment the consistency characteristic of this particular sample.

On the other hand, the data points in Fig. 9.2 are more uniformly spread throughout the axes reflecting inherent inconsistency in the problem–solution spaces. One observes a fair degree of concentration in the lower-left and upper-right quadrants (which suggest comparable values of $DRR_{C,T}$ and $RRR_{C,T}$). However, one also observes many data points in the upper-left and lower-right quadrants indicating instances where target and candidate cases are close in the problem space but far apart in the solution space or vice versa. Of course, such a spread may weaken a user's confidence in the system.

Such visualisations may offer less for samples with high correlation values. For others samples with lower correlation values, such visual impressions of the overall case base reflects presence of some unreliable cases that underscore case base inconsistency and degrade of overall performance. Hence, the next task is to identify such cases.

Figs. 9.3 and 9.4 represent the overall consistency of the same samples, but provide a richer presentation. Both figures plot the $RRR_{C,T}$ from instances in increasing order of distance. In an ideal sample of the BT data set, one would expect to see a uniform transition of paler shaded blocks on the lower-half of the plot to darker blocks on the upper half. Candidate cases in both figures are sorted by their Spearman's Rank correlation values as reported in Table 9.2.

Table 9.2: Case Quality using Spearman's Rank Correlation

BT Sample 6		BT Sample 10	
Case No.	**Correlation**	**Case No.**	**Correlation**
1	0.98	1	0.85
2	0.94	2	0.85
3	0.80	3	0.84
4	0.74	4	0.79
5	0.74	5	0.77
6	0.73	6	0.63
7	0.71	7	0.55
8	0.66	8	0.45
9	0.65	9	0.42
10	0.60	10	0.35
11	0.58	11	0.14
12	0.35	12	-0.25

However, in Fig. 9.3 (representing `sample 6` with correlation coefficient 0.70) substantial number of paler shaded blocks are concentrated on the lower-half of the plot while darker shaded ones are distributed on the upper-half for a large majority of candidate cases. Certainly, this reflects overall consistency and support for the results from Mantel's test since the overall distribution of the coloured blocks is closer to Fig. 6.2. The figure also helps identify suspect unreliable cases such as 11. But it is important to note that in this sample, such cases are very few in number and do not fare so poorly (since the column's lower blocks have relatively paler shades), and this accounts for overall high correlation of the case base. On the other hand, all cases with an exception of `case 6` seem very reliable given the distribution of the coloured blocks in their respective columns in which the paler blocks are consistently concentrated on their lower-halves while the darker are present in their upper-halves.

In contrast, in Fig. 9.4 the sample case base (correlation = 0.3) is observed to be more irregular. This is evident from the random distribution of shaded blocks across the entire breadth of the axes. The lower-half of the plot comprises some dark shaded blocks which denote very poor solution quality given the proximity of target and candidate cases in the problem space. For instance, `cases 10` and `11` are prominently

unreliable since darker blocks are consistently concentrated on the lower-half while paler blocks are present on the upper-half of their columns. Other cases do not serve much better as candidate cases as reflected by the distribution of colours in columns. Hence, overall this sample of the BT data set is inappropriate for CBP and must be pre-processed before use.

Thus, so far in this section, results from Mantel's test have been confirmed by using two extreme samples for demonstration. First a relatively general visualisation was presented that exhibited overall case base consistency and inconsistency. Then a richer visual insight of the regularity in case bases was provided which helped identify suspect (evidence being inadequately objective) unreliable constituent cases. Unfortunately, the visualisation is intended to display only the consistency of only the first 11 cases.

9.1.3 Case Profile

Given the constraints of the visualisation techniques (subjectivity and missing cases), now the focus is upon an objective measure that confirms the unreliable characteristic of the suspect cases. Table 9.2 records the Spearman's Rank Correlation between the respective $DRR_{C,T}$ and $RRR_{C,T}$ for each candidate case in the case base from `BT samples 6` and `10`.

In sample 6, there are only 2 cases with correlations < 0.6. This set of cases overlap the suspect cases we identified from Fig. 9.3. Overall, the case base seems very consistent with 7 cases having correlation > 0.7. These values explain the overall high value of Mantel's test correlation and the desired near-uniform transition of shades in Fig. 9.3. Such a case base is very likely to attract high user confidence. Note that the table also records the correlation for the missing `case 12` in Fig. 9.3.

In addition, the table also confirms the irregular characteristic of `sample 10`. Six constituent cases have correlation values < 0.6 including a case with negative correlation. Of the remaining cases, five have correlation values > 0.7. Thus, in comparison to `sample 6`, this sample of the BT data set has more number of irregular or inconsistent cases and consequently, it seems likely that a poor solution will be delivered by its use.

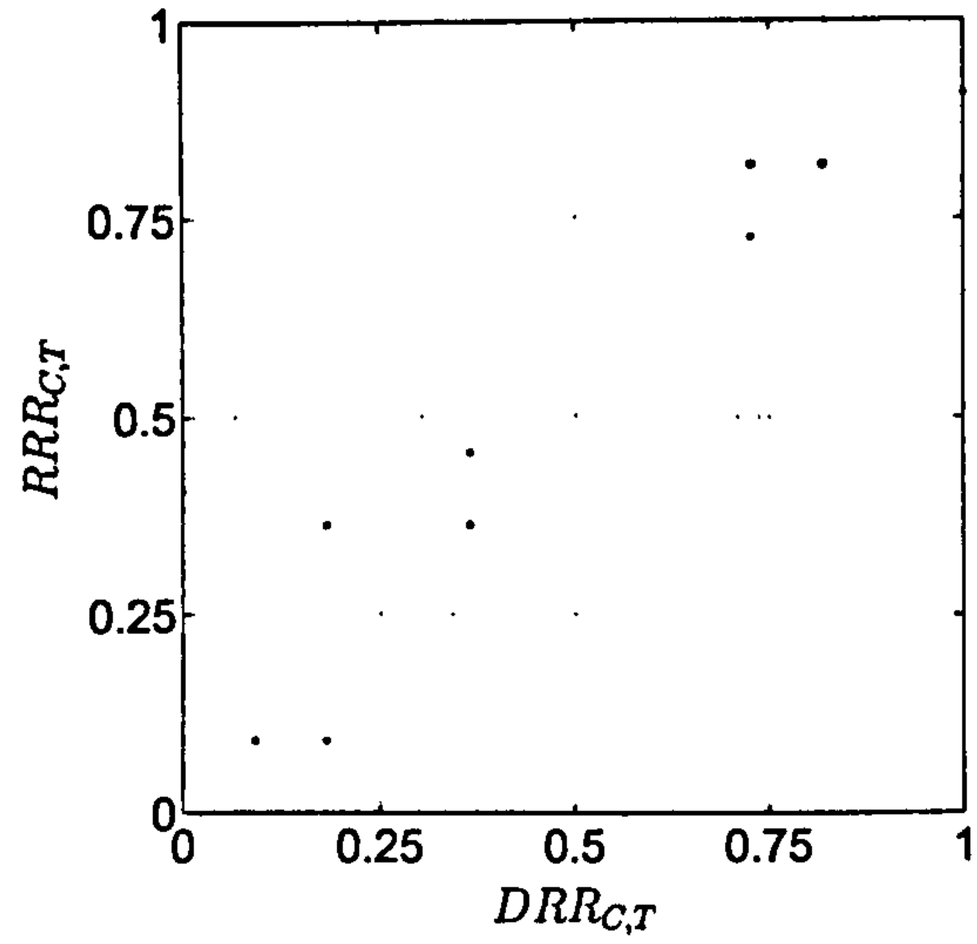

Figure 9.5: Case Profile for Reliable Case 7 from BT Sample 6

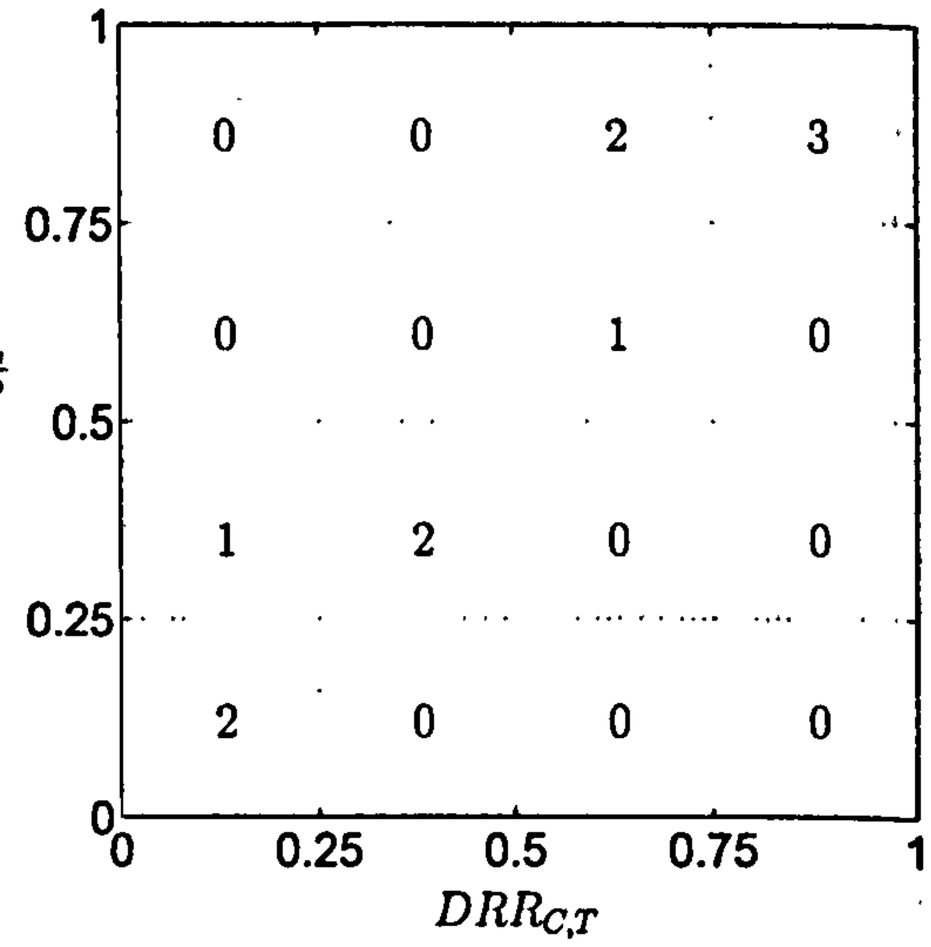

Figure 9.6: Case Profile for Reliable Case 7 from BT Sample 6

Figs. 9.5 and 9.7 demonstrate individual case profiles constructed for a case from each sample (corresponding counts in each quartile are presented in Figs. 9.6 and 9.8 respectively). Figs. 9.5 and 9.6 are examples of a reliable case (case 7 from sample 6) with most of its $DRR_{C,T}$ and $RRR_{C,T}$ being comparable. Such a case is expected to deliver a reliable solution when reused in the future and hence, its usage must be encouraged. On the other hand, Figs. 9.7 and 9.8 are examples of a poor case 4 from sample 10. Data points are seen to be scattered all over its profile matrix. There is a risk of a poor solution being delivered from the use of this case and hence, its usage in the future must be discouraged. It is noteworthy that according to Table 9.2, case 7 from sample 6 has the seventh highest correlation value. This case was chosen simply as a good example of the utility of case profiles. This was less strongly demonstrated by more strongly correlated cases since their data points primarily resided in the top-right quadrant, meaning the cases are distant from other cases in the case base, in both, problem and solution spaces.

9.1.4 Enhancing Prediction

Now, results from using the generated case profile matrices in an attempt to enhance prediction accuracy are presented. The methodology followed has been described in Chapter 6. To summarise, for a given target case, all candidate cases in the case

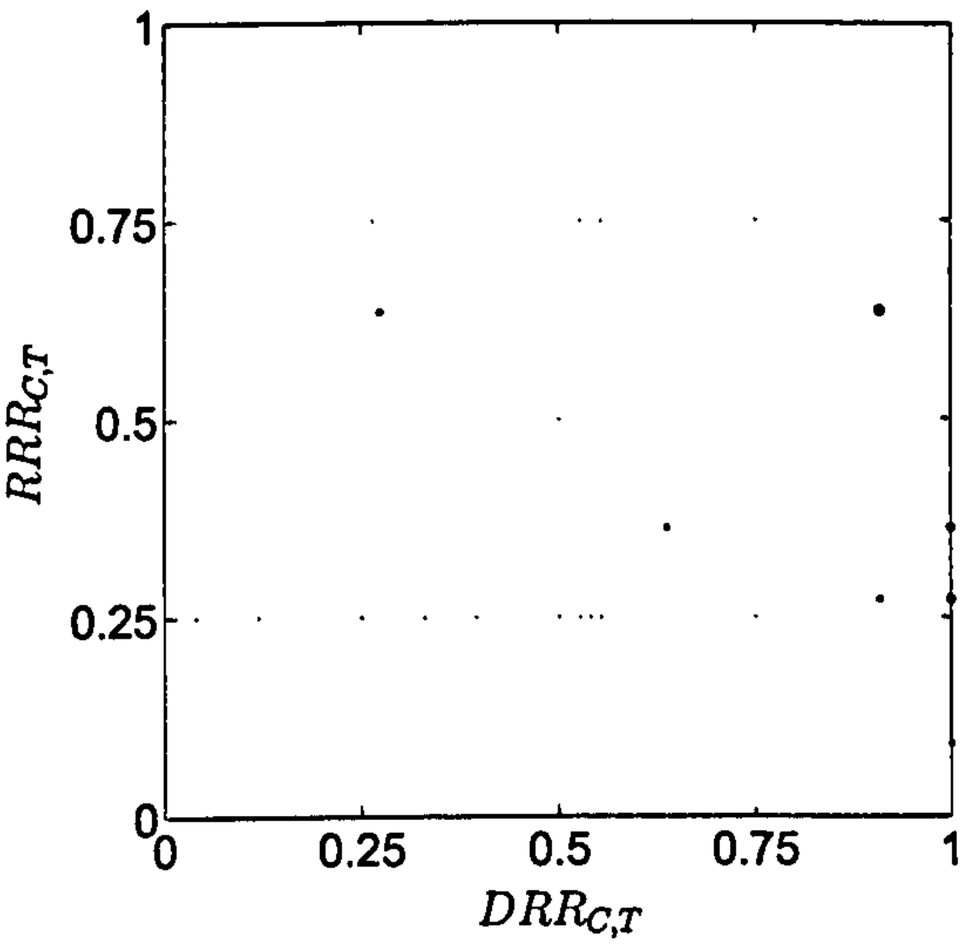

Figure 9.7: Case Profile for Unreliable Case 4 from BT Sample 10

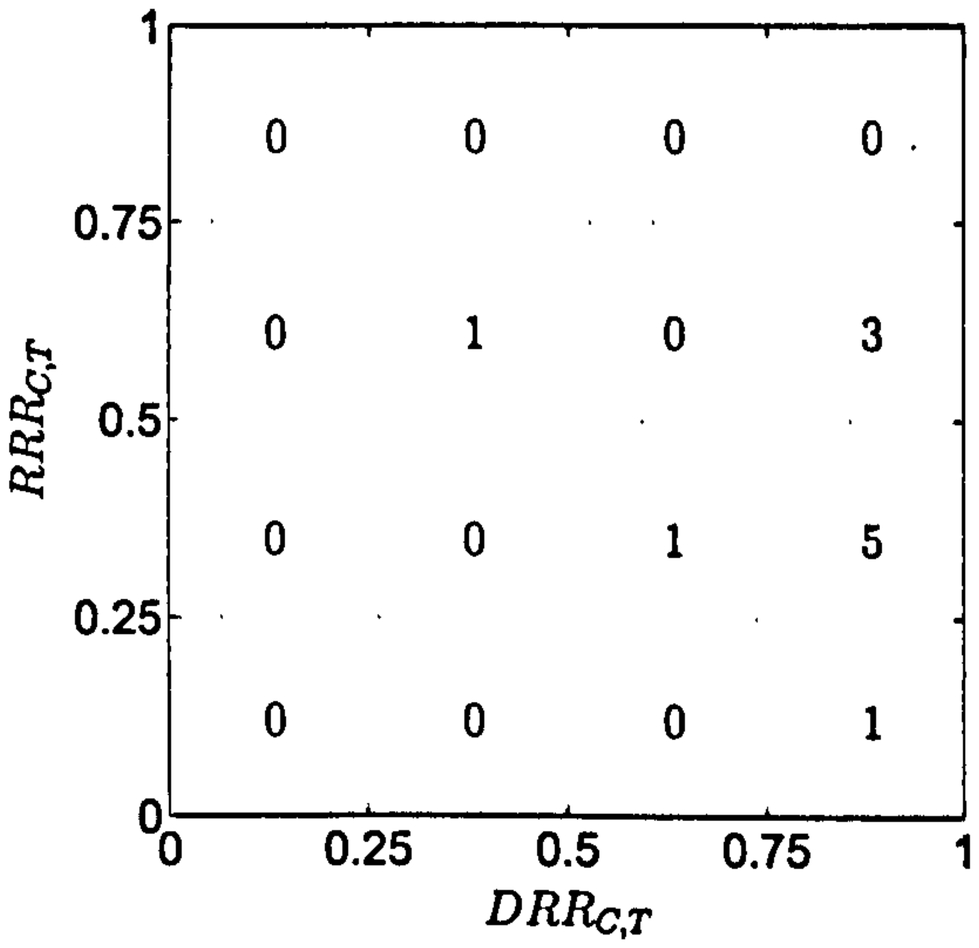

Figure 9.8: Case Profile for Unreliable Case 4 from BT Sample 10

base were sorted in increasing order of Euclidean distance. Thereafter, beginning with the closest candidate case, the probability ($P_{1,2}$) of each case delivering a good solution was computed (separate predictions were to be made using each equation). If a candidate case had $P_{1,2}$ beyond a set threshold limit, the case was reused. Else, the next nearest case that satisfied the performance criteria was used for prediction. This cycle continued until k nearest cases that met the criteria were identified and then, were reused to deliver a solution for the target case.

Unfortunately, for the BT data set, retrieving cases using P_2 largely failed since it required a high frequency in the lower blocks of the individual case profiles. Prediction was successful (for the entire test set) only when using exceptional training samples which commanded a high value of correlation between predictor and response variables. This may therefore be a consequence of the size of the training set and varying inherent irregularity across case bases. Hence, predictions were made by using only P_2. The size of the data set and inherent irregularity also constrained the range of PT (initially intended to be $[0.1 \ldots 0.8]$) to $[0.1 \ldots 0.3]$). These constraints indicate that the applicability of our technique is conditional upon the size and quality of the case base itself and hence, the parameters are required to be determined by trial.

The mean of the $Sum|Res|$ from each of the 30 random testing samples have been

Table 9.3: Mean of $Sum|Res|$ from the BT Data Set

$\frac{k\rightarrow}{Probability\downarrow}$	**1**	**2**	**3**	**4**	**5**
0	850.53	764.40	729.93	740.03	750.27
0.1	836.37	776.40	720.87	741.63	783.33
0.2	836.37	776.40	722.27	744.90	789.77
0.3	836.37	775.97	724.10	757.77	796.00

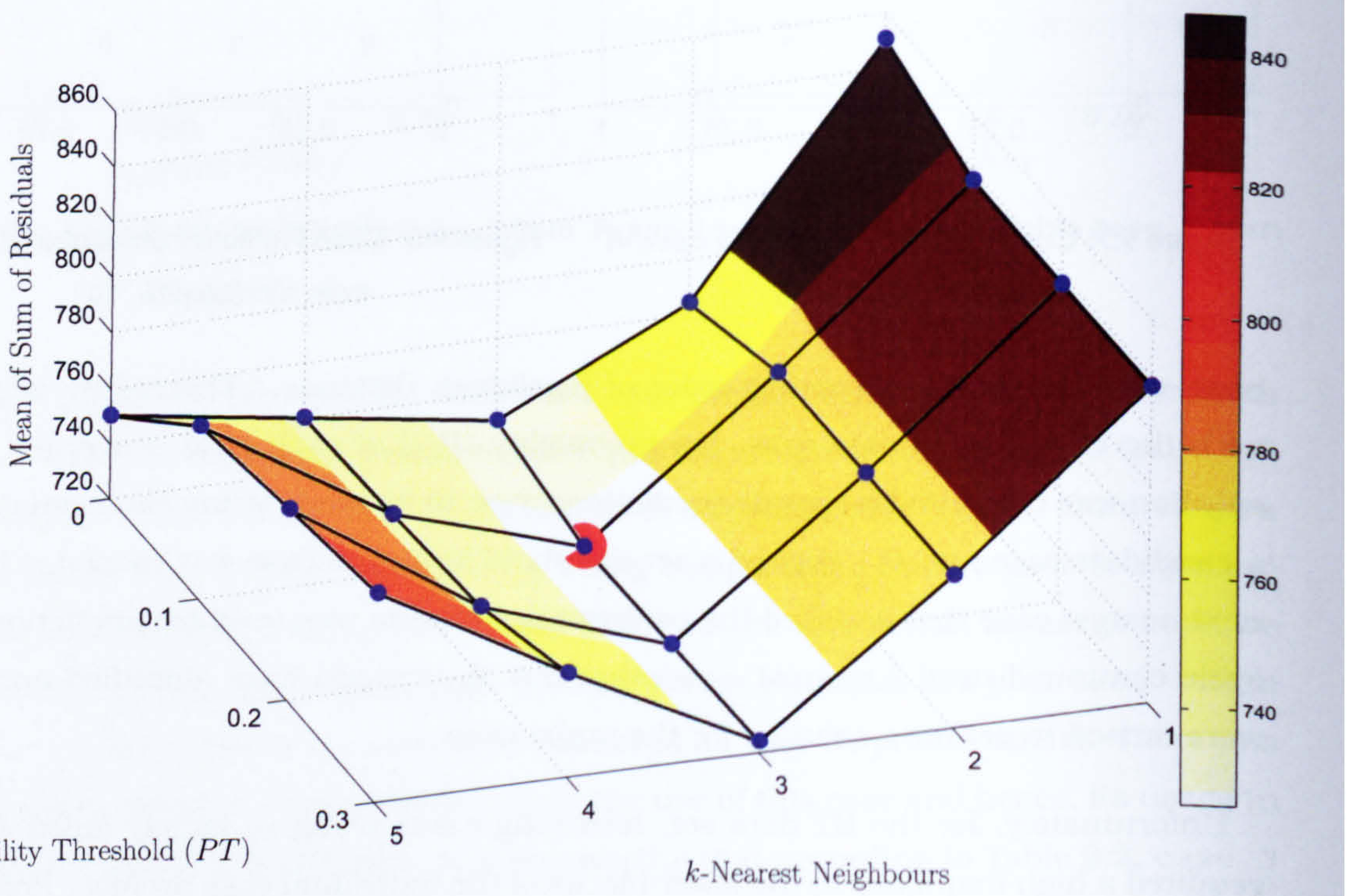

Figure 9.9: Mean of $Sum|Res|$ from BT Data Set

presented in Table 9.3 and plotted in Fig. 9.9. From both, it is observed that the $Sum|Res|$ reduces when using more neighbours until $k = 3$ and thereafter increases. However, once case reliability is gauged using P_2 starting at $PT = 0.1$ before reuse, little overall difference is seen in the $Sum|Res|$. In fact, for $PT = 0.1$, the residuals only decreased in two instances ($k = 1, 3$). Beyond $PT = 0.1$, the residuals seemed to either remain constant or even increase. Despite having a marginal effect upon residuals, the lowest $Sum|Res|$ recorded was by using 3 nearest neighbours along with $PT = 0.1$.

The Kruskal-Wallis test was performed to compare residuals from using only k and those from the combination of k and PT which gave the lowest residuals. For each

case, the null hypothesis could not be rejected. Thus, for the BT data set, using case profiles for selecting candidate cases to reuse makes a modest yet statistically not significant improvement in prediction accuracy. There could be several possible explanations for this including the small size of the test sets. Also, few cases were rejected for use which indicates that the data set was fairly regular to begin with.

9.2 Desharnais Data Set

The Desharnais data set was the second largest data set analysed comprising of 77 complete cases. The training and testing sets comprised 51 and 26 cases respectively. Also, 2550 instances of meta-data were generated for each of the 30 random samples of the data set.

9.2.1 Mantel's Randomisation Test

Similar to the BT data set, first results from the Mantel's Randomisation test (Section 6.1) on the Desharnais data set are examined. For each of the 30 samples, the correlation between the original pair of distance matrices was computed. Thereafter, another 4999 correlation coefficients were calculated between the original residual matrix and 4999 randomisations of the distance matrix. Results have been recorded in Table 9.4 which includes the original correlation coefficient, maximum and minimum values from the entire set of 5000 coefficients for each sample.

Again, the results unfolded several interesting characteristics of the data set with some similar to the BT data set. Firstly, the correlation coefficient for each of the 30 random samples between the original distance matrices was positive and thus, it encourages the use of this data set for CBP.

Secondly, although positive, the value of correlation from each sample was consistently low (unlike the BT data set). The highest recorded value was 0.34 for `Sample 5` and the lowest was 0.15 for `Sample 14`. The weak strength of correlation suggests the existence of many outliers that contribute towards overall irregularity in the case base. The belief was further strengthened by the range and low variance of the correlation values which imply that every random sample contained at least few unreliable cases

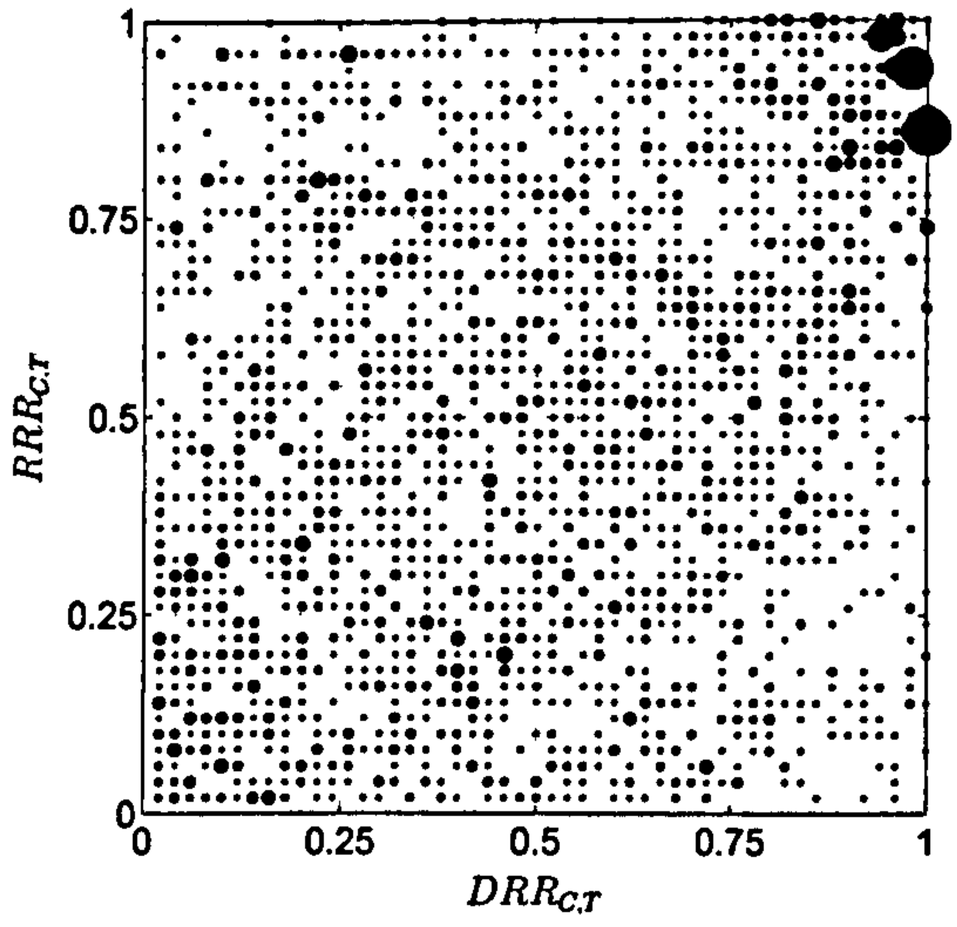

Figure 9.10: $DRR_{C,T}$ vs. $RRR_{C,T}$ for Regular Desharnais Sample 5

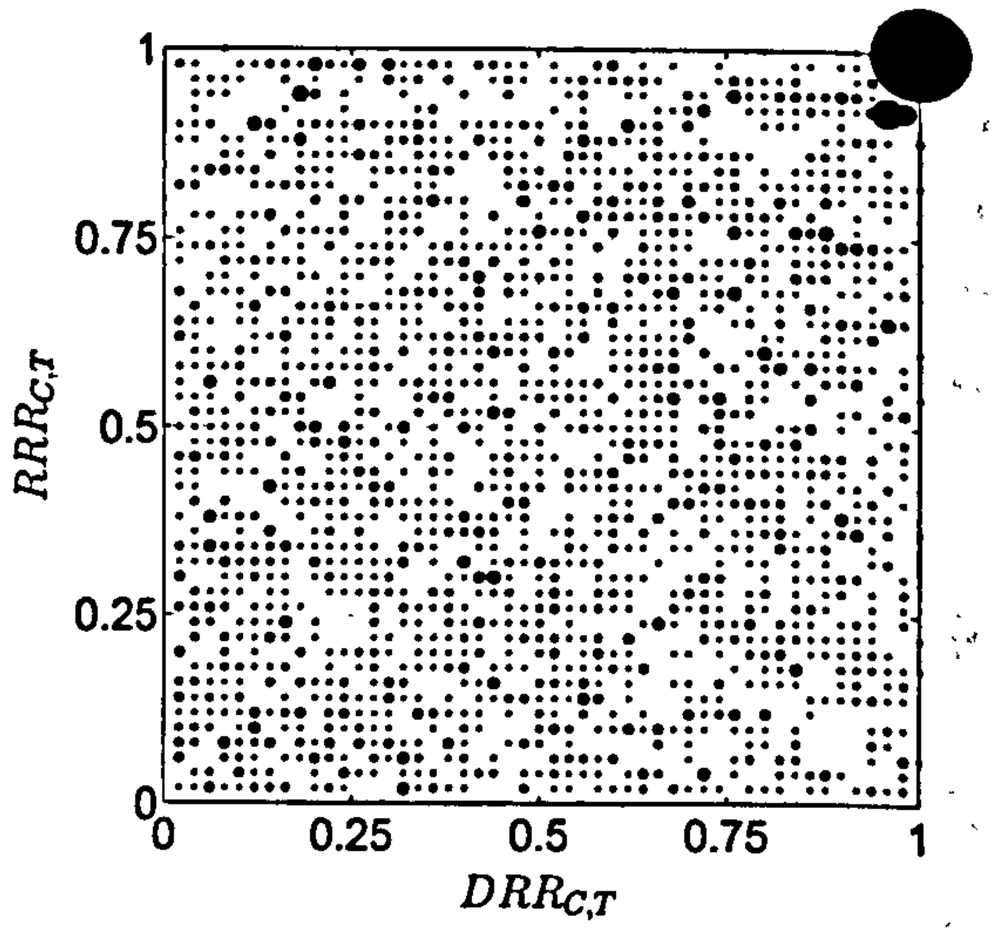

Figure 9.11: $DRR_{C,T}$ vs. $RRR_{C,T}$ for Irregular Desharnais Sample 14

that distorted overall irregularity. Hence, overall for this data set, there is a need to supplement inter-case distance with more information prior to selecting the case for reuse.

Lastly, for each random sample, the correlation coefficient between the original distance matrices was the highest amongst all 5000 computed coefficients. Thus, each sample passed the test of statistical significance ($p < 0.001$). This is an important observation which verifies that there indeed exists a pattern between the predictor and response variables or confirms a problem–solution relationship. Thus, any prediction model or method would derive the best possible results using the original pairs of predictor and response variables.

Hence, though the positive values and test of significance support its use for CBP, the Desharnais data set appears to have a larger number of unreliable cases given the low correlation values for each of the 30 samples. This cautions us to make a more informed decision about which cases to select for reuse in order to increase prediction accuracy.

Table 9.4: Mantel's Randomisation Test Results on the Desharnais Data Set

Sample		Corr.	Sample		Corr.	Sample		Corr.
	Max.	0.25		**Max.**	0.29		**Max.**	0.25
1	**Original**	0.25	**11**	**Original**	0.29	**21**	**Original**	0.25
	Min	-0.07		**Min**	-0.06		**Min**	-0.07
	Max.	0.30		**Max.**	0.25		**Max.**	0.22
2	**Original**	0.30	**12**	**Original**	0.25	**22**	**Original**	0.22
	Min	-0.06		**Min**	-0.05		**Min**	-0.06
	Max.	0.30		**Max.**	0.30		**Max.**	0.25
3	**Original**	0.30	**13**	**Original**	0.30	**23**	**Original**	0.25
	Min	-0.06		**Min**	-0.06		**Min**	-0.06
	Max.	0.23		**Max.**	0.15		**Max.**	0.20
4	**Original**	0.23	**14**	**Original**	0.15	**24**	**Original**	0.20
	Min	-0.07		**Min**	-0.06		**Min**	-0.06
	Max.	0.34		**Max.**	0.25		**Max.**	0.32
5	**Original**	0.34	**15**	**Original**	0.25	**25**	**Original**	0.32
	Min	-0.06		**Min**	-0.07		**Min**	-0.06
	Max.	0.25		**Max.**	0.18		**Max.**	0.23
6	**Original**	0.25	**16**	**Original**	0.18	**26**	**Original**	0.23
	Min	-0.07		**Min**	-0.07		**Min**	-0.06
	Max.	0.25		**Max.**	0.21		**Max.**	0.23
7	**Original**	0.25	**17**	**Original**	0.21	**27**	**Original**	0.23
	Min	-0.06		**Min**	-0.06		**Min**	-0.06
	Max.	0.32		**Max.**	0.23		**Max.**	0.21
8	**Original**	0.32	**18**	**Original**	0.23	**28**	**Original**	0.21
	Min	-0.07		**Min**	-0.06		**Min**	-0.05
	Max.	0.34		**Max.**	0.25		**Max.**	0.23
9	**Original**	0.34	**19**	**Original**	0.25	**29**	**Original**	0.23
	Min	-0.07		**Min**	-0.06		**Min**	-0.07
	Max.	0.29		**Max.**	0.24		**Max.**	0.21
10	**Original**	0.29	**20**	**Original**	0.24	**30**	**Original**	0.21
	Min	-0.07		**Min**	-0.06		**Min**	-0.06

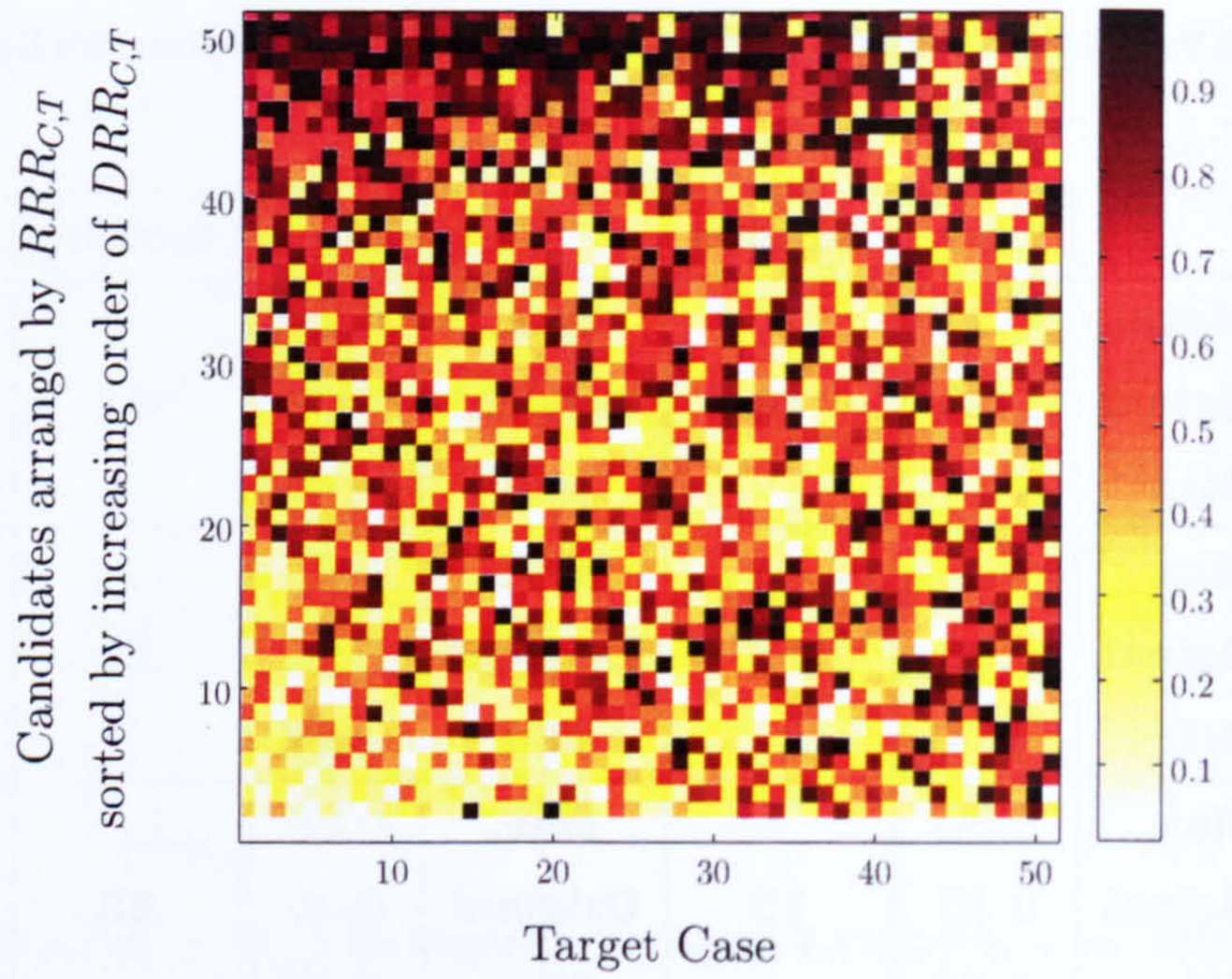

Figure 9.12: $RRR_{C,T}$ sorted by $DRR_{C,T}$ for Regular Desharnais Sample 5

9.2.2 Visualisation Case Base Quality

The overall dissonance in the two extreme samples 5 and 14 is visualised in Figs. 9.10 and 9.11 respectively. The density of data points in the lower-left and upper-right quadrants of the former figure is much higher than in the other two quadrants. Considerable presence of data points in upper-left and lower-right quadrants may be an atrefact of the number of cases in the case base (and hence the large number of meta-data instances) and the degree of inherent irregularity (correlation = 0.34). The presence of bigger data points along the diagonal of the figure further supports its regularity.

On the other hand, Fig. 9.11 plots the meta-data instances from `sample 14` having a correlation coefficient as low as 0.15. It is evident from the distribution of the data points that this sample has more inherent irregularity in comparison to `sample 5`. While many of the data points lie in the lower-left and upper-right quadrants, in comparison to `sample 5` there seem to be more points in the upper-left and lower-right quadrants giving evidence of frequent poor instances of reuse. Additionally, the presence of darker data points in the latter quadrants further explain the resultant low correlation value and make evident the inherent irregularity.

In Figs. 9.12 and 9.13, the same two samples are again represented in an attempt

Table 9.5: Case Quality using Spearman's Rank Correlation for `Desharnais Sample 5`

Case	Correlation	Case	Correlation	Case	Correlation
1	0.71	18	0.44	35	0.29
2	0.69	19	0.43	36	0.29
3	0.68	20	0.42	37	0.28
4	0.68	21	0.41	38	0.28
5	0.65	22	0.38	39	0.28
6	0.65	23	0.38	40	0.27
7	0.64	24	0.38	41	0.25
8	0.58	25	0.37	42	0.25
9	0.57	26	0.37	43	0.22
10	0.55	27	0.36	44	0.16
11	0.55	28	0.36	45	0.15
12	0.53	29	0.35	46	0.14
13	0.50	30	0.34	47	0.13
14	0.47	31	0.33	48	0.12
15	0.47	32	0.31	49	0.03
16	0.46	33	0.30	50	-0.01
17	0.45	34	0.29	51	-0.10

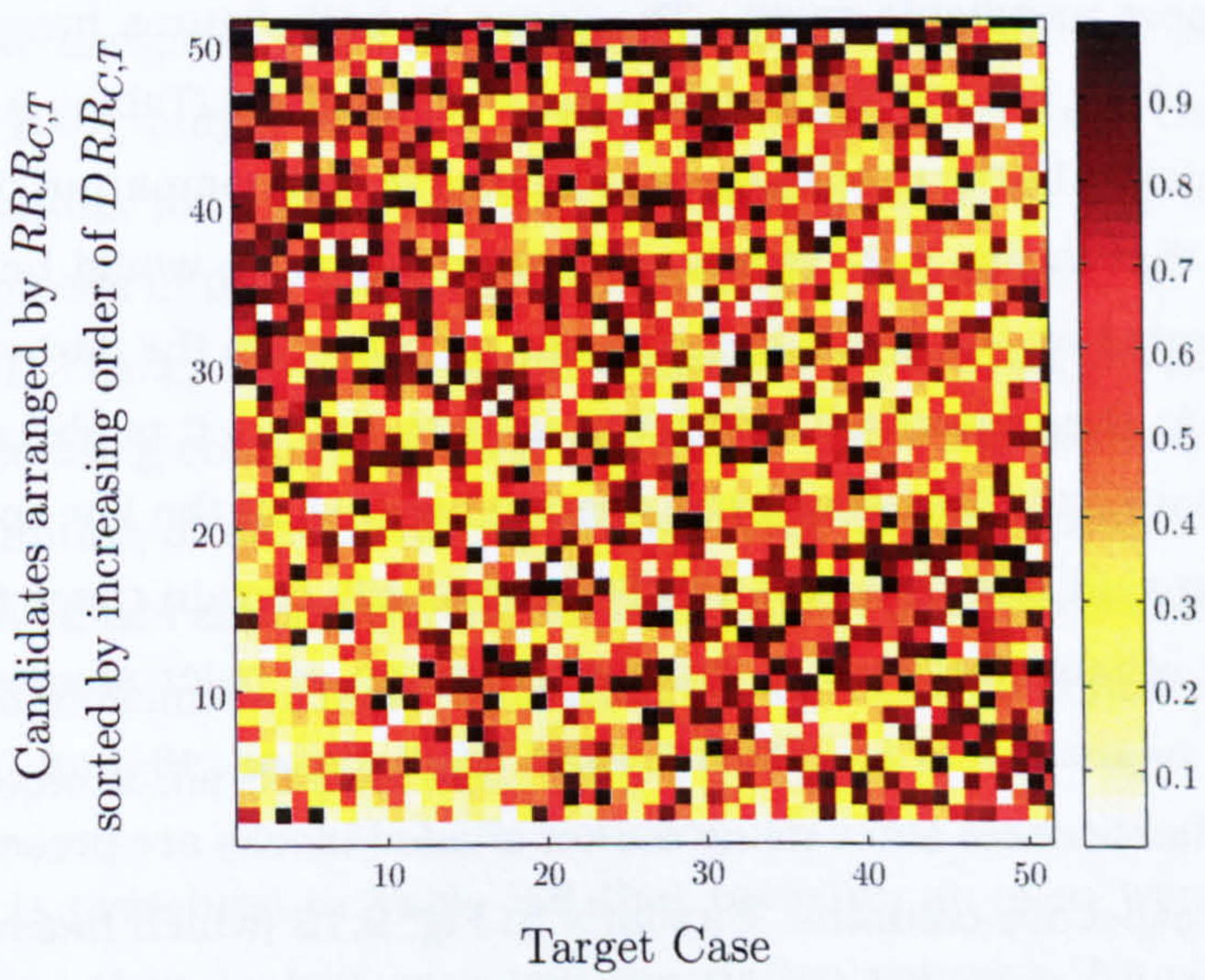

Figure 9.13: $RRR_{C,T}$ sorted by $DRR_{C,T}$ for Irregular Desharnais Sample 14

Table 9.6: Case Quality using Spearman's Rank Correlation for `Desharnais Sample 14`

Case	Correlation	Case	Correlation	Case	Correlation
1	0.65	18	0.24	35	0.11
2	0.51	19	0.24	36	0.10
3	0.48	20	0.24	37	0.10
4	0.45	21	0.22	38	0.09
5	0.43	22	0.22	39	0.09
6	0.43	23	0.22	40	0.09
7	0.40	24	0.21	41	0.06
8	0.39	25	0.20	42	0.05
9	0.38	26	0.20	43	0.04
10	0.37	27	0.18	44	0.03
11	0.36	28	0.16	45	0.02
12	0.35	29	0.16	46	0.01
13	0.34	30	0.16	47	-0.04
14	0.34	31	0.16	48	-0.05
15	0.32	32	0.15	49	-0.09
16	0.30	33	0.15	50	-0.11
17	0.28	34	0.12	51	-0.47

to identify suspect unreliable cases. The cases in both figures have been sorted by the descending order of their Spearman's rank correlations (Tables 9.5 and 9.6). The degree of irregularity in the case base can be observed by comparing their distribution of shades with that in Fig. 6.2. Ideally, a regular case base would have paler shaded blocks concentrated on the lower-half of each column while the darker ones would be largely present in the upper-half. However, in both figures it is observed that this is not the case. Both samples seem to be chaotic because of the low correlation values for each. Having said this, it is still possible to identify certain cases in Fig. 9.12 that can be deemed reliable because of the concentration of paler shaded blocks on the lower-half, e.g. `cases 10 to 50`. On the other hand, `cases 48 and 50` are examples of suspect unreliable cases since many darker shaded blocks are present on the lower-halves of their respective columns. Similarly in Fig. 9.13 (which has a relatively more random distribution of blocks), `cases 8 to 18` seem be be examples of the reliable cases while `cases 44 to 49` exhibit unreliable behaviour.

Hence, like for the BT data set, Figs. 9.12 and 9.13 demonstrate the benefits of using different visualisation techniques on the Desharnais data set to judge the overall quality of the case base. However, the two extreme samples examined visually seem more or less equally random given their low correlation values, thus also validating the results of Mantel's test. Again, unfortunately, the latter pair of case base visualisations restrain the number of cases plotted to one less than the total number of cases in the case base.

9.2.3 Case Profile

The visualisation methods demonstrated above are very effective to quickly glance at the incoherence existing in the case base. However, besides being rather subjective, another important limitation brought to light by the Desharnais data set is the effect of larger case bases. Having more cases results in a dense visualisation that makes identifying unreliable cases more tedious and perhaps for larger case bases nearly impossible by manual scrutiny.

In Tables 9.5 and 9.6, the Spearman's rank correlations are presented for cases in `samples 5` and `14` of the Desharnais data set. To recall, the correlation values record the strength of association between the $DRR_{C,T}$ and $RRR_{C,T}$ for each candidate case in the case base.

Out of the two samples, `5` has the higher correlation value of 0.34 and is represented by Table 9.5. Here, only 7 of the 51 candidate cases have a correlation value > 0.6. Respective columns for these cases can be cross-checked in Fig. 9.12, each having paler shaded blocks in their lower halves and darker blocks on their upper halves. On the other hand, 8 cases have correlation values < 0.2 including 2 with negative values. Again, corresponding columns of these unreliable cases are observed to have a rather random distribution of shaded blocks in Fig. 9.12. A larger proportion of cases with low correlation values explain the overall low value of Mantel's correlation for the case base. It is also indicative of more constituent cases being unreliable in comparison to the better sample of the BT data set.

`Sample 14` is presented in Table 9.6 that presents an even worse case base with correlation 0.15. Here, only 1 case had correlation values > 0.6 while 25 cases had correlation < 0.2. Thus, nearly half the case base is comprised of unreliable cases.

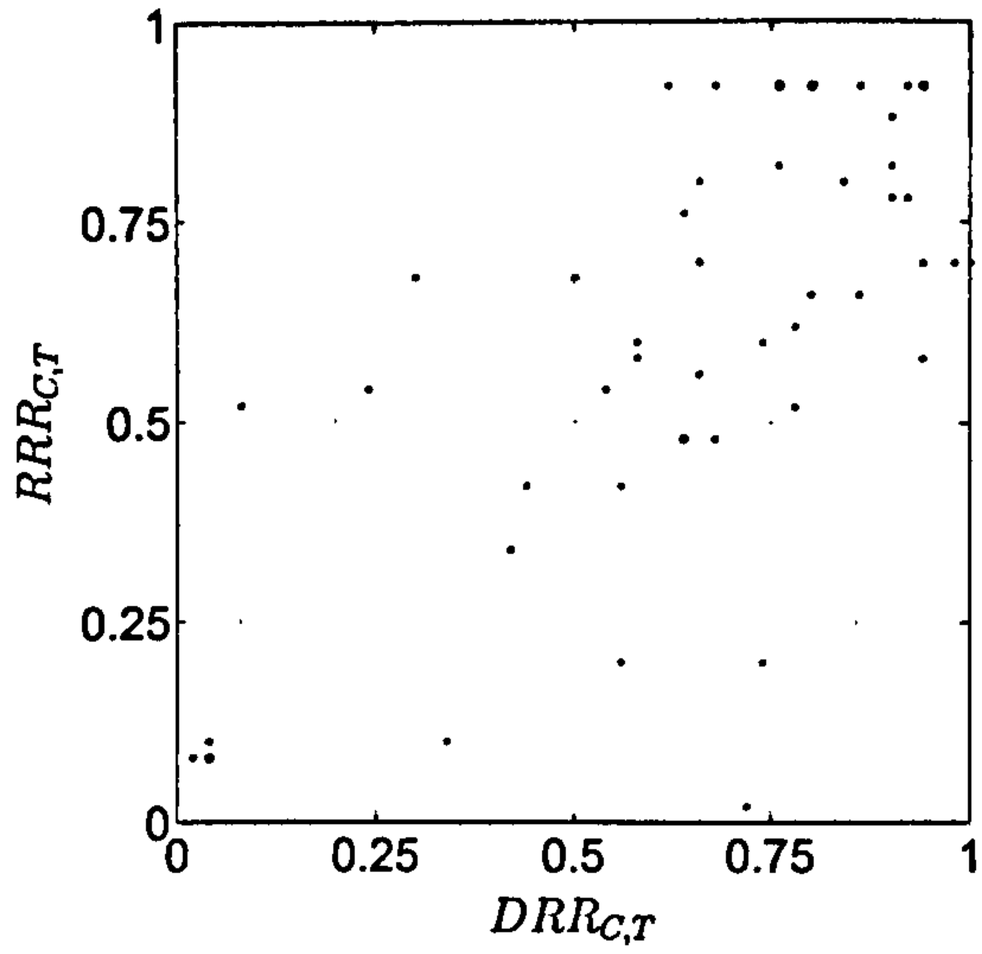

Figure 9.14: Case Profile for Case 49 from Desharnais Sample 5

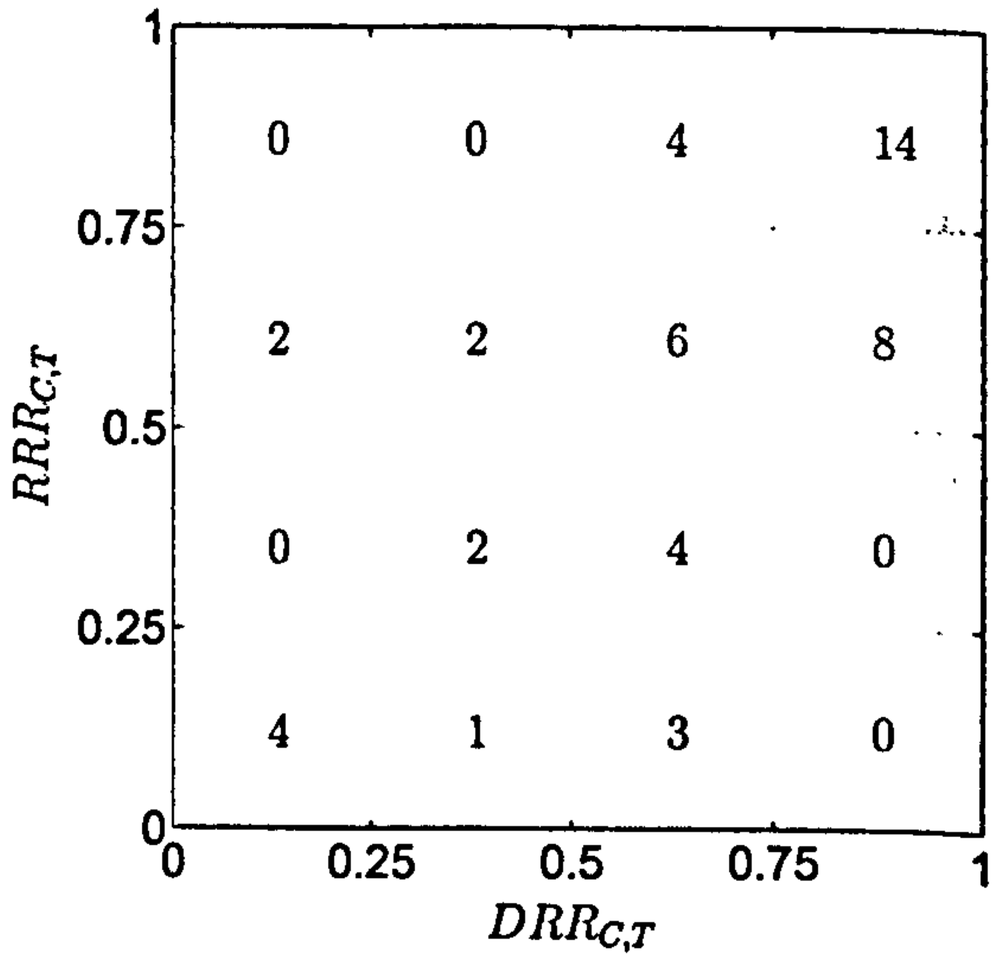

Figure 9.15: Case Profile for Case 49 from Desharnais Sample 5

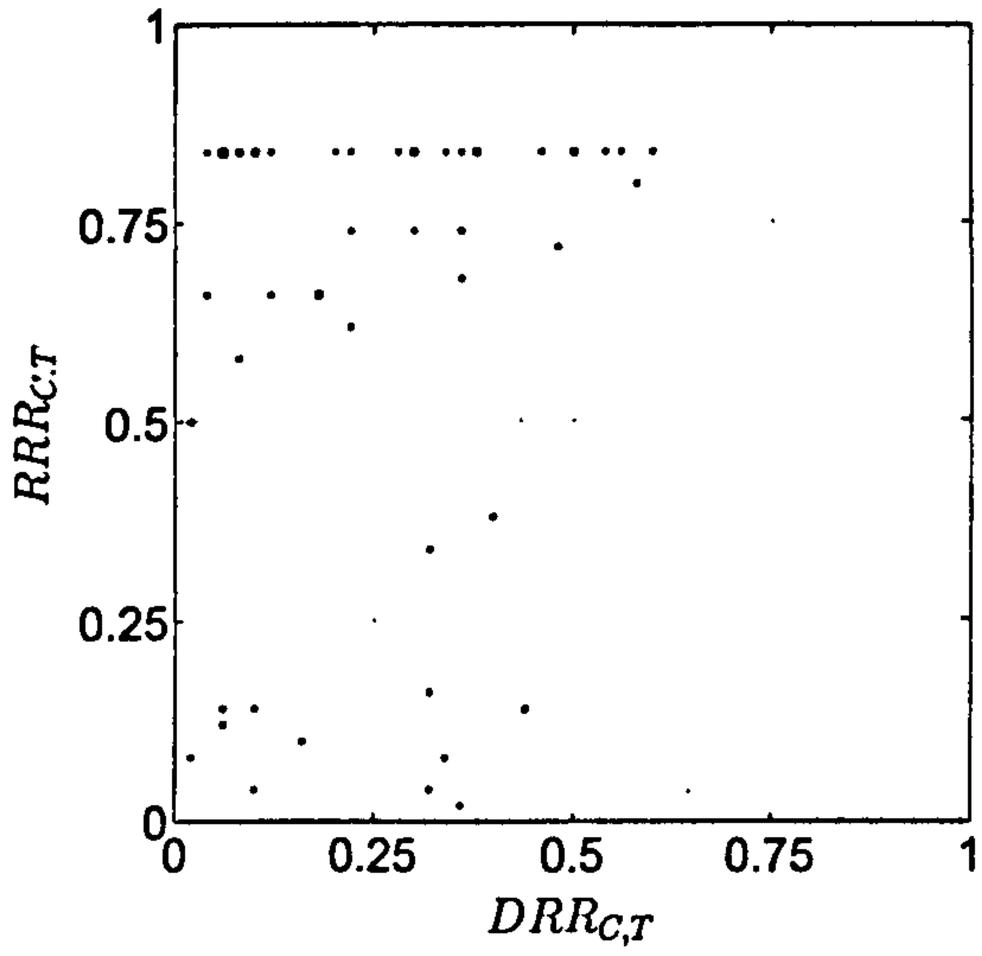

Figure 9.16: Case Profile for Case 39 from Desharnais Sample 14

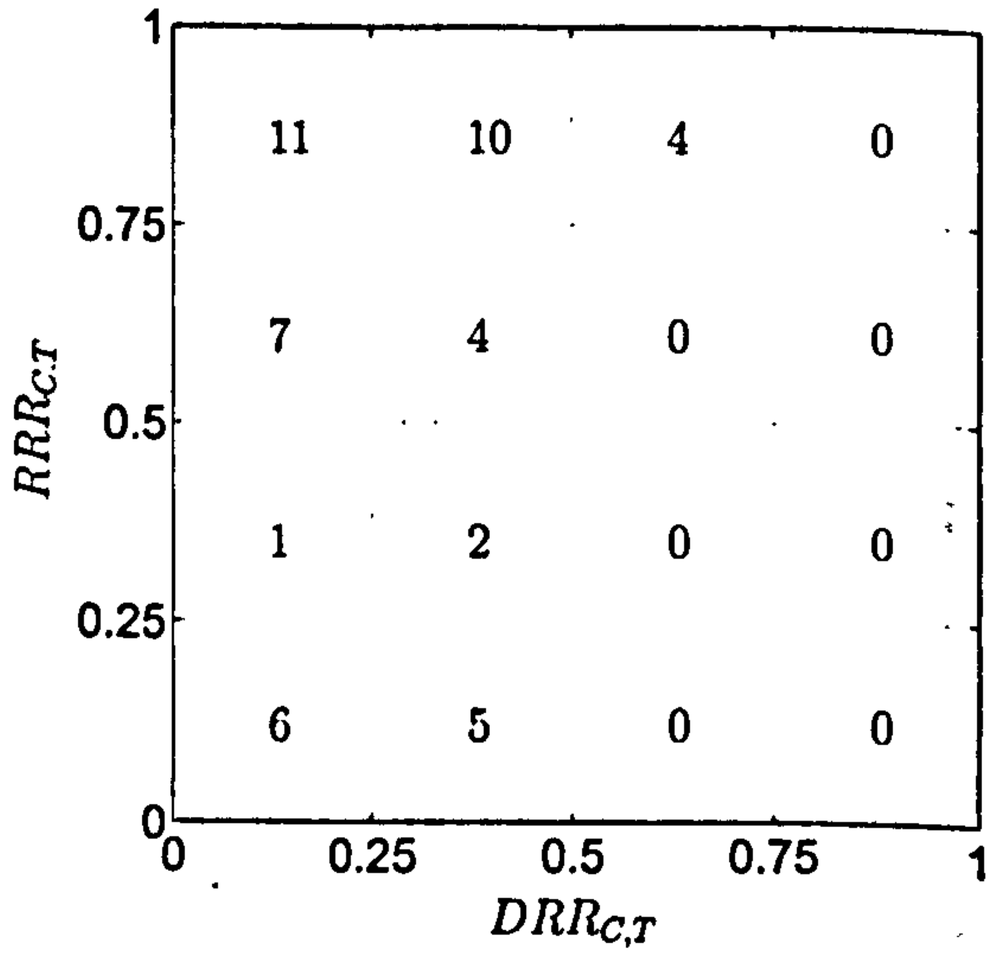

Figure 9.17: Case Profile for Case 39 from Desharnais Sample 14

Again, the corresponding columns of the cases can be cross-checked in Fig. 9.13 that exhibit either uniformity or irregularity in the distribution of shaded blocks. Resultantly, overall this case base is likely to deliver poorer solutions in comparison to the other sample but the difference would most likely be inconsiderable given the weak correlations for both training sets.

In Figs. 9.14 and 9.16, individual case profiles have been demonstrated. The former figure is an example of a reliable case (case 49 from sample 5) where most of the data

points are concentrated across comparable values of $DRR_{C,T}$ and $RRR_{C,T}$. Interestingly, the few times this case has been very close to the target (low values of $DRR_{C,T}$), it has provided a good solution. Otherwise, this case is dominantly distant from most of the other cases in the case base (hence the dense cloud in the upper-right quadrants) and as expected, provides distant solutions. The distribution of the data points across the $DRR_{C,T}$ also highlights that this case is rather unique in both, problem and solution spaces. Hence, it appears to reliably cover a certain portion of the domain.

On the other hand, in Fig. 9.14 we demonstrate (`case 39` from `sample 14`) as an unreliable case which is evident from the numerous data points in the upper-left quadrants of the plot. The points indicate close proximity of `case 39` to the target cases in the problem space, yet they are very distant in the solution space. Hence, such a case should be reused with extreme caution, especially considering how similar it is to most other cases in the case base.

9.2.4 Prediction

This section presents the impact of combining case quality information with distance on prediction accuracy using the Desharnais data set. We aimed to measure case reliability using both, P_1 and P_2. However, again retrieving cases using P_2 failed due to the inherent inconsistency in all samples. As a result, predictions were made across using all combinations of $k([1 \ldots 5])$ and $PT([0 \ldots 0.8])$, but only using P_2. Fig. 9.18 is a plot of the mean of $Sum|Res|$ from each of the 30 samples and the respective values are recorded in Table 9.7.

We first examine the behaviour using k-NN exclusively for selecting cases for reuse i.e. $PT = 0$. Using only the nearest neighbour ($k = 1$) results in the largest $Sum|Res|$. However, the sum of absolute residuals continue to decline as k increases to 5 (accuracy increasing by $\approx 17\%$). During the experiments on this data set, it was found that often, k-NN retrieved cases that were lying far away on opposite sides of the target case in the solution space and thus, provided a ballpark solution by averaging the extreme values. Resultantly, a larger value of k lessened the effect of extreme values on the proposed solution. Such a pattern is also in line with some previous research [KCS00] where an increase in k (up to a limit) neutralised the effect of outliers and resultantly

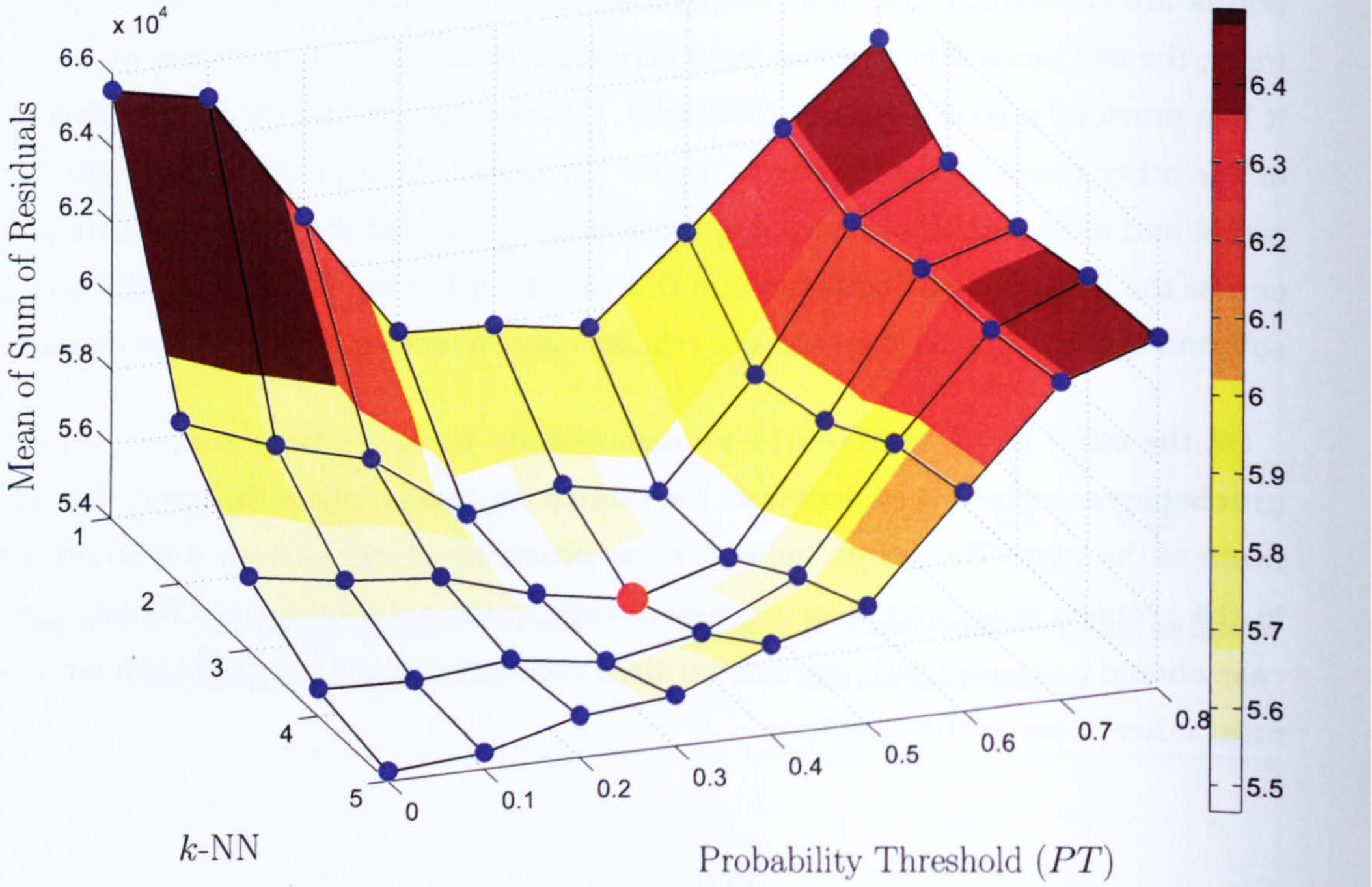

Figure 9.18: Comparison of Performance (Means) by Coupling k-NN and Probability Threshold for the Desharnais Dataset

reduced the $Sum|Res|$. Though the solution may potentially be close to the true value, this technique is likely to reduce the confidence of users in the system.

However, once coupled with PT set even as low as 0.1, we observe an improvement in performance for every value of k other than 5. Barring $k = 5$, the trend of improved performance continued until $PT = 0.3$ and thereafter, the $Sum|Res|$ again began increasing. This is because the system became more discriminating and overlooked many similar cases in search of high quality cases. As a result, very distant cases got reused and this decreased prediction accuracy. For this data set, we found the optimum combination of k and PT was 3 and 0.4 respectively giving the lowest means $Sum|Res|$. A total gain of 17.4% improvement in prediction accuracy was made from using more than one nearest neighbour and probability threshold in comparison to using the nearest neighbour only.

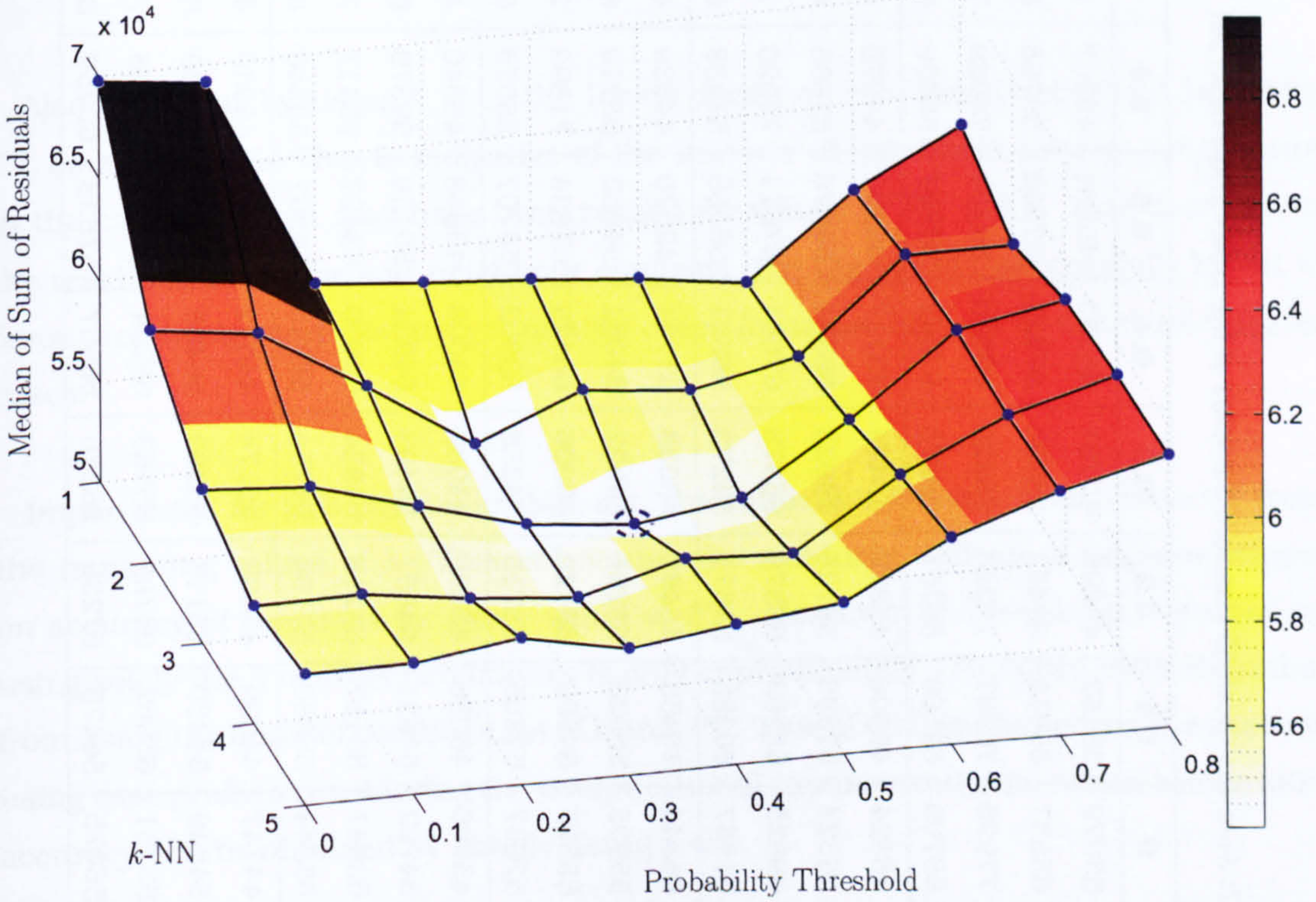

Figure 9.19: Comparison of Performance (Medians) by Coupling k-NN and Probability Threshold for the Desharnais Dataset

Table 9.7: $Sum|Res|$ from 30 Random Samples of the Desharnais Data Set

$\frac{Probability\rightarrow}{k\downarrow}$		0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
1	**LB**	53079	53157	49698	47781	46153	45284	45507	48405	50975
	Mean	65537	65012	61462	58050	57922	57532	59768	62245	64340
	UB	77995	76867	73226	68320	69690	69781	74029	76085	77705
	Median	69709	69109	58757	58273	57942	57547	56870	60790	63476
2	**LB**	46824	46506	45877	43474	43117	42141	44070	47291	48447
	Mean	58324	57385	56725	54923	55384	54876	57695	61508	62799
	UB	69825	68263	67574	66373	67651	67611	71320	75724	77152
	Median	61547	60858	57671	54273	56521	56069	57238	61678	61671
3	**LB**	46344	46456	45466	43038	41734	42070	44539	47618	48661
	Mean	55933	55528	55294	54522	54081	54852	58219	62002	62811
	UB	65523	64600	65122	66006	66429	67634	71899	76385	76962
	Median	57607	57205	55718	54338	53902	54707	58075	62018	63055
4	**LB**	45796	46056	45496	44032	43008	43848	45340	48100	48869
	Mean	54722	54621	54857	54493	54954	56138	59379	62119	63249
	UB	63647	63187	64219	64954	66800	68427	73417	76138	77629
	Median	55844	55880	55164	54754	56215	55957	59346	61857	63415
5	**LB**	44747	44899	45437	45174	44542	44503	45939	48593	48982
	Mean	54249	54430	55113	55369	56408	57109	59839	62485	63379
	UB	63751	63962	64788	65563	68274	69715	73738	76376	77777
	Median	56475	56475	57236	56276	56967	57528	60259	62114	63497

Table 9.8: Kruskal-Wallis Test Results for Desharnais Data Set

k	PT	p-value
1	0.5	< 0.05
2	0.5	< 0.05
3	0.4	< 0.05
4	0.3	0.08
5	N/A	N/A

Also note that by using $k = 1$, the lowest mean of $Sum|Res|$ is obtained by using PT as high as 0.5. This is indicative of the number of unreliable nearest neighbours in the case base that may have been reused for values of $PT < 0.5$. This decrease in the residuals using the sole neighbour confirms that the system successfully learnt to favour relatively more distant yet quality cases for reuse instead of unreliable nearer cases.

Importantly, for this particular data set, the value of $k = 5$ stood out distinctly from the remaining values of k. This is because the residuals indicate a negative impact on accuracy of solutions for every value of PT. Moreover, the residuals obtained by using solely the 5 nearest neighbours is only approximately 170 hours more than that from using the optimal combination of k and PT. This of course challenges the worth of using case profiles considering the computational expense and time, when comparable accuracy can be obtained by simply using $k = 5$.

Hence, we investigated further by observing the median of $Sum|Res|$ plotted in Fig. 9.19. By using medians too, $k = 3$ and $PT = 0.4$ gave the lowest sum of residuals. However, there was a much larger difference by using this combination of k and PT in comparison with using only the nearest neighbour since the error reduces by approximately 22%. Also, setting PT even as low as 0.1 induced a marginal increase in prediction accuracy. The accuracy tends to increase for values of $PT = 0.6$ and then began increasing due to reuse of more distant cases. We also noticed a larger difference in the median values between the optimal combination of k and PT and $k = 5$. Thus, the comparable prediction accuracy using mean of $Sum|Res|$ may be an artefact of some extreme values negating each other.

Thus, both Figs. 9.18 and 9.19 indicate that prediction accuracy increases by avoiding reuse of unreliable similar cases using the proposed case discrimination system. To further confirm this, results from the Kruskal-Wallis test have been presented in Table 9.8. The first column indicates the value of k while the second column indicates the value of PT for which the $Sum|Res|$ was least when used along with the corresponding value of k. Lastly, the third column records the attained p-value from the tests. The results were significant for $k = 1, 2, 3$ indicating the $Sum|Res|$ using k along with the proposed case discrimination system were significantly lower (at $\alpha = 5\%$) than reusing the k nearest neighbours. This is an important result since it statistically verifies the value of the technique. and in these three cases, the null hypothesis can be rejected. However, for $k = 4$, the results were not significantly different indicating that higher values of k tend to neutralise extreme outliers, which are seemingly avoided from reuse by the case discrimination system for lower values of k. Hence, the null hypothesis could not be rejected. Lastly, for $k = 5$, the lowest $Sum|Res|$ were obtained using k alone, but now lower than the optimal results achieved using $k = 3$ and $PT = 0.4$.

9.3 Finnish Data Set

The Finnish data set was the largest of all three data sets used in this research. With a total of 207 remaining cases after cleaning the data set (Section 8.3), in each of the 30 random samples, the training set comprised 137 cases while the remaining 70 cases constituted the testing set. As a result, 18,632 (i.e. $137 * 136$) instances of meta-data were generated for each sample.

9.3.1 Mantel Test

Again, analysis of the Finnish data set begins with the Mantel test, results for which have been recorded in Table 9.9. For each of the 30 random samples, the correlation between the original pair of distance matrices was computed. Thereafter, another 4999 correlation coefficients were calculated between the original residual matrix and 4999 randomisations of the distance matrix. Results in Table 9.9 include the origi-

nal correlation coefficient, maximum and minimum values from the entire set of 5000 coefficients for each sample.

The value of correlation for each sample was positive meaning that there is a tendency in the data set for points in the problem and solution spaces to move in the same direction. Of course, this is an important starting criteria to be satisfied in order to justify to use this data set for CBP.

However, the values of correlation for each sample were quite low. `Sample 19` had the highest correlation value of 0.18 while `sample 9` had the lowest value of 0.07. Interestingly, only 5 samples had a correlation value < 0.1 while the remaining 25 samples had values distributed between 0.1 and 0.18. Such small variance in the correlation values strongly suggests a high degree of irregularity in each training set due to presence of large number of influential unreliable cases. Beyond doubt, this gives a strong signal that any case base comprising such cases must be either pre-processed or dealt with extreme caution during reuse.

Similar to the Desharnais data set, for all 30 samples of the Finnish data set, R_1 was greater than the other correlation values between the randomised and original distance matrices. Thus each sample passed the test of significance at $p < 0.001$. This suggests a true (albeit weak) relationship in the real world between the problem space features and the solution features. Not only does this result help explain casuality, it also suggests that any model built upon empirical data is likely to perform best using the original combination of problem and solution space features.

9.3.2 Visualisation Case Base Quality

We now turn to visualising the case base irregularity for `samples 19` and `9` in Figs. 9.20 and 9.21. The former figure depicts the training set with correlation 0.18 and is extremely densely distributed with data points across the entire axes indicating many instances of good and poor reuse. But the marginally less density of points in the upper-left and lower-right quadrants is noteworthy, especially in relation to Fig. 9.21, and larger bubbles are concentrated around the diagonal. Thus, despite being the best or most highly correlated of all samples, we observe a very high degree of dissonance in the sample which would almost certainly influence solution quality.

Table 9.9: Mantel's Randomisation Test Results on the Finnish Data Set

Sample		Corr	Sample		Corr	Sample		Corr
	Max.	0.14		**Max.**	0.12		**Max.**	0.13
1	**Original**	0.14	**11**	**Original**	0.12	**21**	**Original**	0.13
	Min.	-0.02		**Min.**	-0.03		**Min.**	-0.02
	Max.	0.15		**Max.**	0.12		**Max.**	0.15
2	**Original**	0.15	**12**	**Original**	0.12	**22**	**Original**	0.15
	Min.	-0.02		**Min.**	-0.02		**Min.**	-0.02
	Max.	0.12		**Max.**	0.15		**Max.**	0.11
3	**Original**	0.12	**13**	**Original**	0.15	**23**	**Original**	0.11
	Min.	-0.03		**Min.**	-0.02		**Min.**	-0.02
	Max.	0.09		**Max.**	0.14		**Max.**	0.12
4	**Original**	0.09	**14**	**Original**	0.14	**24**	**Original**	0.12
	Min.	-0.02		**Min.**	-0.03		**Min.**	-0.02
	Max.	0.13		**Max.**	0.15		**Max.**	0.15
5	**Original**	0.13	**15**	**Original**	0.15	**25**	**Original**	0.15
	Min.	-0.02		**Min.**	-0.02		**Min.**	-0.02
	Max.	0.15		**Max.**	0.08		**Max.**	0.12
6	**Original**	0.15	**16**	**Original**	0.08	**26**	**Original**	0.12
	Min.	-0.02		**Min.**	-0.02		**Min.**	-0.02
	Max.	0.12		**Max.**	0.09		**Max.**	0.12
7	**Original**	0.12	**17**	**Original**	0.09	**27**	**Original**	0.12
	Min.	-0.03		**Min.**	-0.02		**Min.**	-0.02
	Max.	0.09		**Max.**	0.12		**Max.**	0.12
8	**Original**	0.09	**18**	**Original**	0.12	**28**	**Original**	0.12
	Min.	-0.02		**Min.**	-0.02		**Min.**	-0.02
	Max.	0.07		**Max.**	0.18		**Max.**	0.11
9	**Original**	0.07	**19**	**Original**	0.18	**29**	**Original**	0.11
	Min.	-0.02		**Min.**	-0.02		**Min.**	-0.02
	Max.	0.14		**Max.**	0.12		**Max.**	0.15
10	**Original**	0.14	**20**	**Original**	0.12	**30**	**Original**	0.15
	Min.	-0.02		**Min.**	-0.03		**Min.**	-0.02

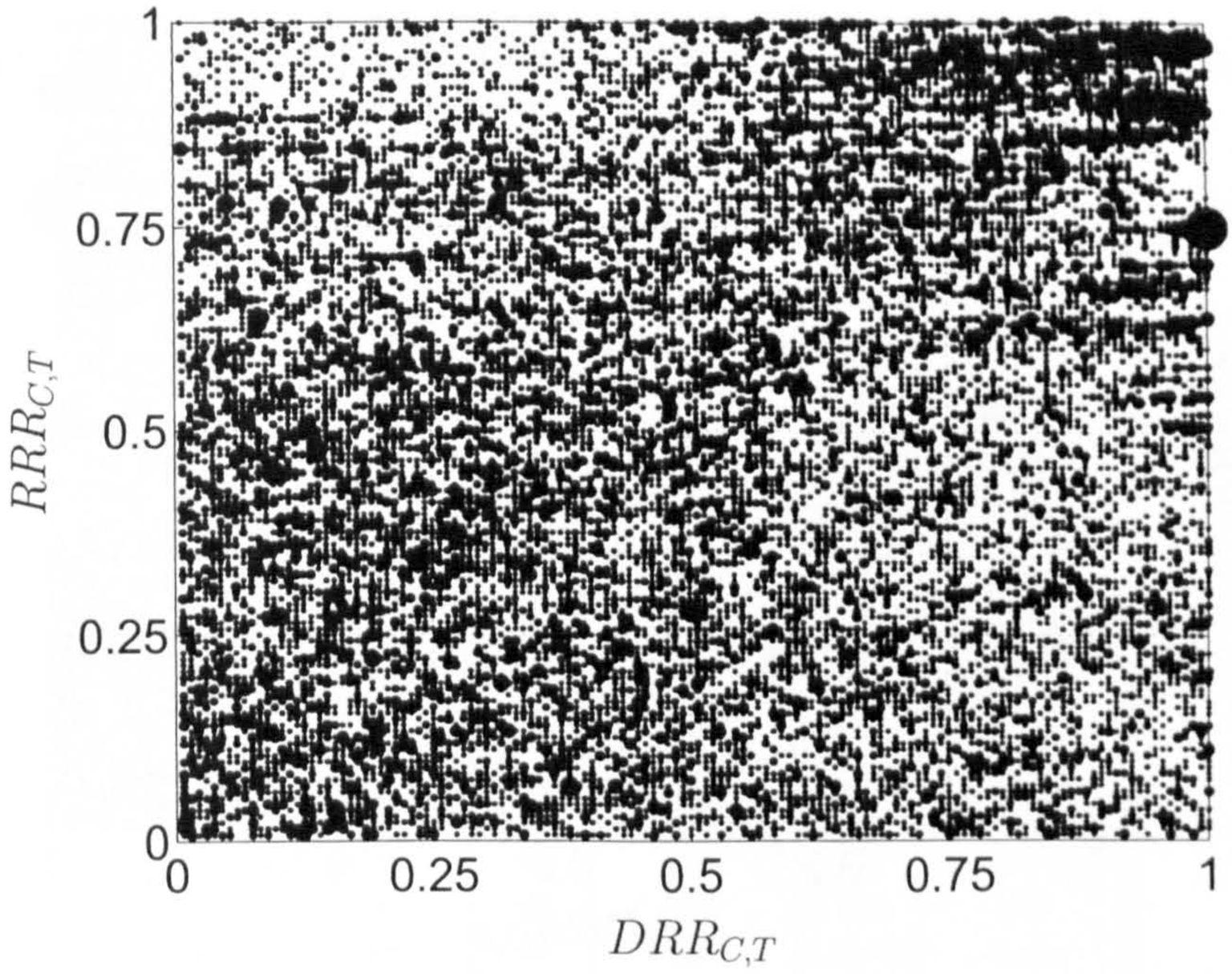

Figure 9.20: $DRR_{C,T}$ vs. $RRR_{C,T}$ for Finnish Sample 19

Fig. 9.21, which depicts the sample with correlation 0.07, is practically little different from the other considering its weak correlation. This is a result of a large number of number of meta-data instances and high irregularity inherent in the case base. The distribution confirms our deductions from the Mantel test that it is imperative to cautiously reuse cases from this data set in order to improve prediction accuracy.

It would appear that both, Figs. 9.20 and 9.21, emphasize that the dissonance in the two case bases is comparable; a direct consequence of low correlation values. Importantly, the figures highlight the limitation of using the bubble plots to visualise case base dissonance. As the case base grows larger, the plot becomes more dense, making any meaningful interpretation from the plots challenging.

To analyse the two samples in more depth, in Figs. 9.22 and 9.23 the meta-data has been plotted to attempt identifying potentially unreliable cases within the same samples 19 and 9. The cases in both plots are arranged in accordance with Tables 9.10

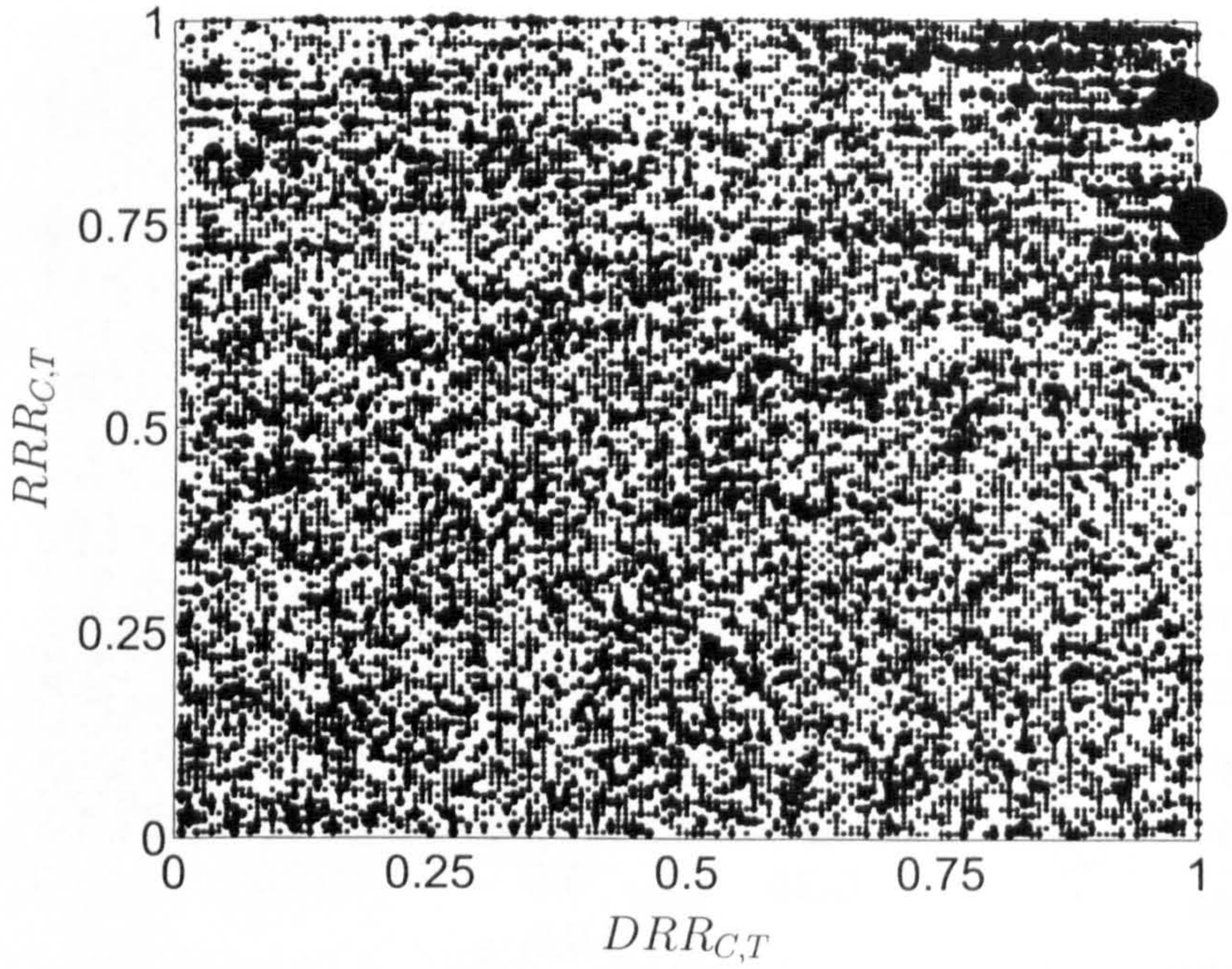

Figure 9.21: $DRR_{C,T}$ **vs.** $RRR_{C,T}$ **for Finnish Sample 9**

and 9.11. Although overall, both figures appear equally random (a result of comparable and low correlations), there are relatively more paler shaded blocks in the lower half of Fig. 9.22 and darker in the upper-half. However, the sheer number of meta-data instances dampen this characteristic of the better sample resulting in a low correlation value for the training set. Though it is possible to invest much time to identify suspect unreliable cases in the two samples, both figures demonstrate the weakness of the visualisation. Such manual identification is virtually impossible for even larger case bases.

An interesting observation in both, Tables 9.10 and 9.11 is the high correlation values for individual cases. Upon verification, it was learnt that most cases in the sample were far away from each other in both problem and solution spaces. As a result, their $DRR_{C,T}$ and $RRR_{C,T}$ had high values resulting in stronger correlation.

Table 9.10: Case Quality using Spearman's Rank Correlation for `Finnish Sample 19`

Case	Correlation	Case	Correlation	Case	Correlation
1	1.00	47	0.70	93	0.30
2	1.00	48	0.70	94	0.30
3	1.00	49	0.60	95	0.20
4	1.00	50	0.60	96	0.20
5	1.00	51	0.60	97	0.20
6	0.90	52	0.60	98	0.20
7	0.90	53	0.60	99	0.20
8	0.90	54	0.60	100	0.10
9	0.90	55	0.60	101	0.10
10	0.90	56	0.60	102	0.10
11	0.90	57	0.60	103	0.10
12	0.90	58	0.60	104	0.10
13	0.90	59	0.60	105	0.10
14	0.90	60	0.60	106	0.10
15	0.90	61	0.60	107	0.10
16	0.90	62	0.60	108	0.10
17	0.90	63	0.50	109	0.10
18	0.80	64	0.50	110	0.10
19	0.80	65	0.50	111	0.00
20	0.18	66	0.50	112	0.00
21	0.80	67	0.50	113	0.00
22	0.70	68	0.50	114	0.00
23	0.70	69	0.50	115	-0.10
24	0.70	70	0.50	116	-0.10
25	0.70	71	0.50	117	-0.10
26	0.70	72	0.50	118	-0.10
27	0.70	73	0.50	119	-0.10
28	0.70	74	0.50	120	-0.20
29	0.70	75	0.50	121	-0.20
30	0.70	76	0.50	122	-0.20
31	0.70	77	0.40	123	-0.20
32	0.70	78	0.40	124	-0.20
33	0.70	79	0.40	125	-0.30
34	0.70	80	0.40	126	-0.30
35	0.70	81	0.40	127	-0.30
36	0.70	82	0.40	128	-0.30
37	0.70	83	0.40	129	-0.50
38	0.70	84	0.40	130	-0.50
39	0.70	85	0.40	131	-0.50
40	0.70	86	0.40	132	-0.60
41	0.70	87	0.40	133	-0.60
42	0.70	88	0.30	134	-0.60
43	0.70	89	0.30	135	-0.60
44	0.70	90	0.30	136	-0.60
45	0.70	91	0.30	137	-0.90
46	0.70	92	0.30		

Table 9.11: Case Quality using Spearman's Rank Correlation for `Finnish Sample 9`

Case	Correlation	Case	Correlation	Case	Correlation
1	1.00	47	0.60	93	-0.30
2	0.90	48	0.60	94	-0.40
3	0.90	49	0.60	95	-0.50
4	0.90	50	0.60	96	-0.50
5	0.90	51	0.60	97	-0.50
6	0.90	52	0.60	98	-0.50
7	0.90	53	0.60	99	-0.60
8	0.90	54	0.60	100	-0.60
9	0.90	55	0.60	101	-0.60
10	0.90	56	0.50	102	-0.60
11	0.80	57	0.50	103	-0.60
12	0.80	58	0.50	104	-0.60
13	0.80	59	0.50	105	-0.60
14	0.80	60	0.40	106	-0.60
15	0.80	61	0.40	107	-0.60
16	0.80	62	0.30	108	-0.60
17	0.80	63	0.30	109	-0.60
18	0.80	64	0.30	110	-0.60
19	0.80	65	0.30	111	-0.60
20	0.80	66	0.30	112	-0.60
21	0.80	67	0.30	113	-0.60
22	0.80	68	0.30	114	-0.60
23	0.80	69	0.30	115	-0.70
24	0.80	70	0.30	116	-0.70
25	0.70	71	0.30	117	-0.70
26	0.70	72	0.20	118	-0.70
27	0.70	73	0.20	119	-0.70
28	0.70	74	0.10	120	-0.70
29	0.70	75	0.10	121	-0.80
30	0.70	76	0.00	122	-0.80
31	0.70	77	0.00	123	-0.80
32	0.70	78	0.00	124	-0.80
33	0.70	79	0.00	125	-0.80
34	0.70	80	-0.10	126	-0.80
35	0.60	81	-0.10	127	-0.80
36	0.60	82	-0.10	128	-0.80
37	0.60	83	-0.10	129	-0.80
38	0.60	84	-0.20	130	-0.80
39	0.60	85	-0.20	131	-0.80
40	0.60	86	-0.20	132	-0.80
41	0.60	87	-0.30	133	-0.80
42	0.60	88	-0.30	134	-0.90
43	0.60	89	-0.30	135	-0.90
44	0.60	90	-0.30	136	-0.90
45	0.60	91	-0.30	137	-0.90
46	0.60	92	-0.30		

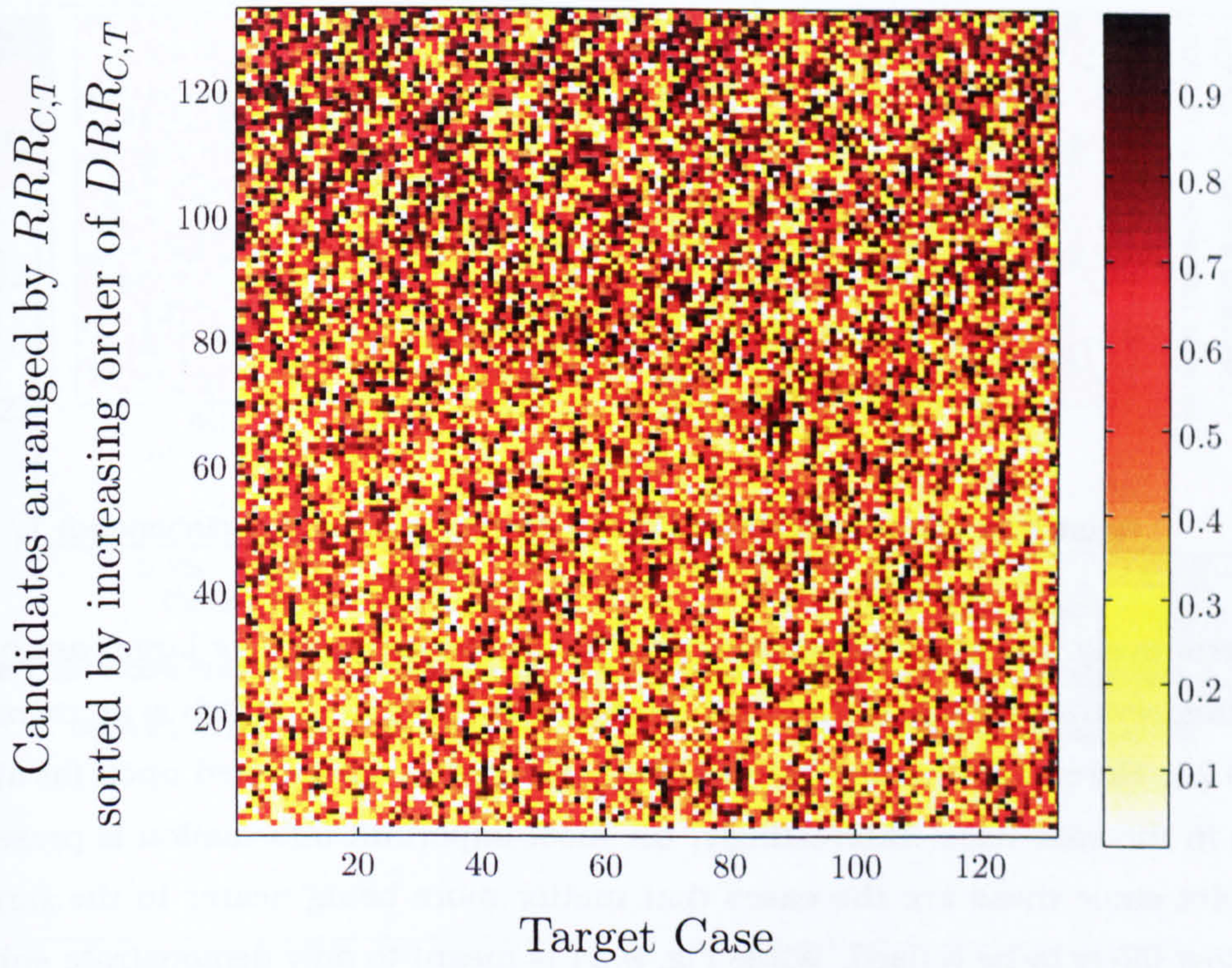

Figure 9.22: $RRR_{C,T}$ sorted by $DRR_{C,T}$ for Finnish Sample 19

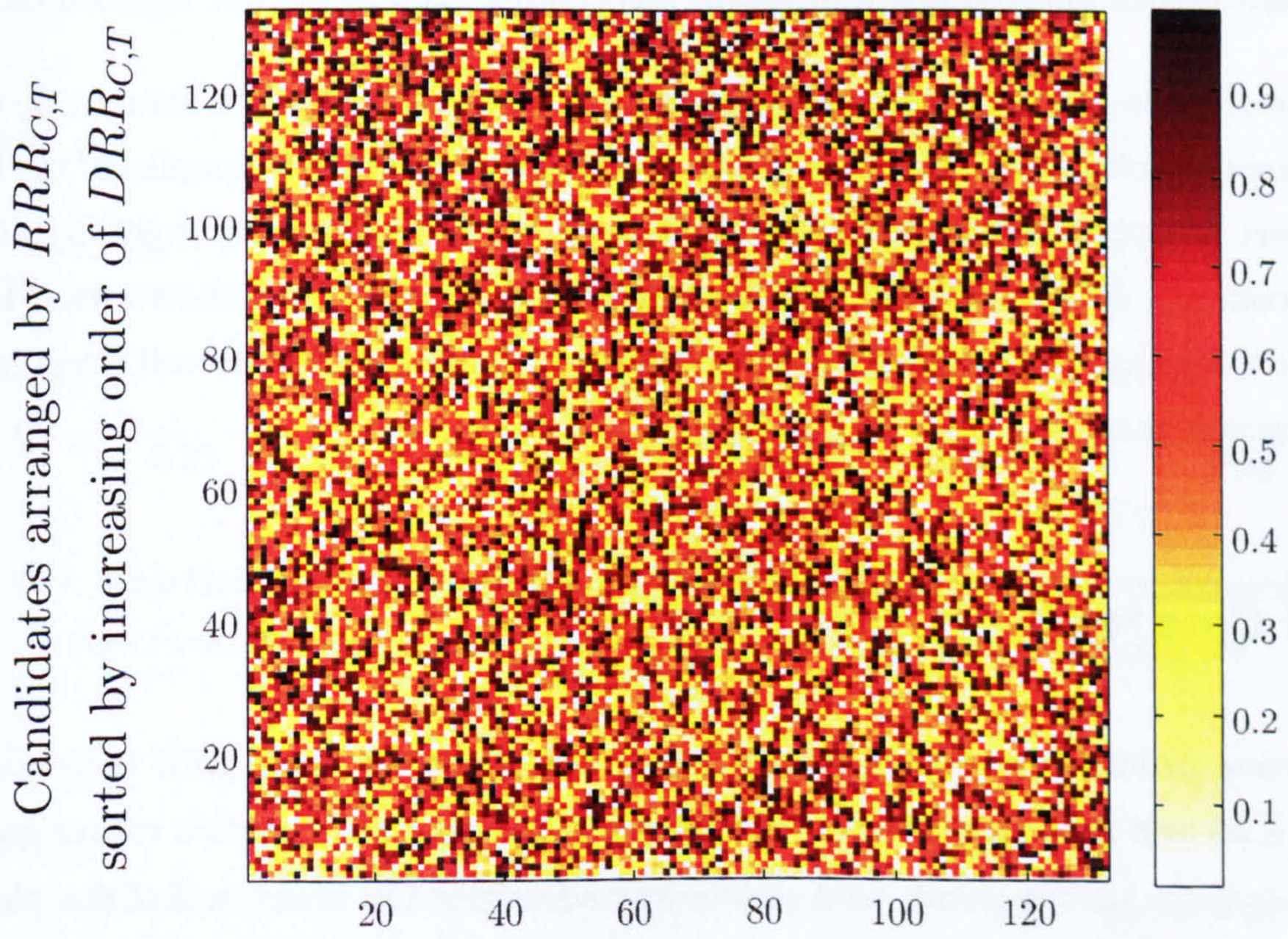

Figure 9.23: $RRR_{C,T}$ sorted by $DRR_{C,T}$ for Finnish Sample 9

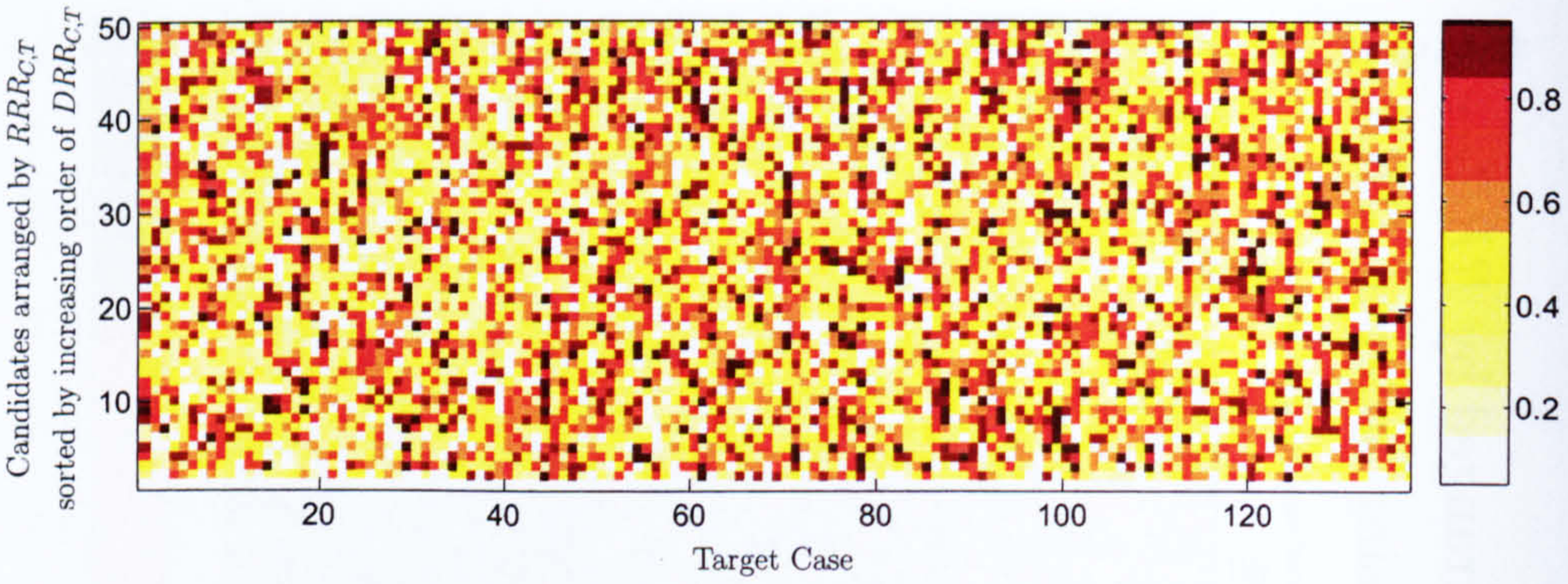

Figure 9.24: $RRR_{C,T}$ sorted by $DRR_{C,T}$ for Finnish Sample 19 (Enhanced)

Alternatively, these visualisations could be further enhanced for large case bases by adopting several possible techniques. An example is Fig. 9.24 which is an extract from Fig. 9.22. Here, only the first 50 candidate cases are concentrated upon for all target cases in the case base. Interestingly, the most important information is preserved in this plot since these are the cases that matter more being nearer to the target and are most likely to be reused. While Fig. 9.24 is meant to only demonstrate enhancing visualisation, other possibilities to accomplish the same include averaging across k different values and using optimisation techniques such as genetic algorithms.

Thus, in this section, results from the Mantel test were confirmed visually by demonstrating the substantial irregularity even in the better of the two samples of the Finnish data set examined. Unfortunately, the respective figures also strongly highlight the shortcomings of the proposed visualisation technique which has shown itself to be ineffective for any further practical use other than examining the overall irregularity for large case bases.

9.3.3 Case Profile

Two case profiles are visualised in Figs. 9.25 and 9.27 along with their values in Figs. 9.26 and 9.28 respectively. The former figure that represents a rather good case from `sample 19` has many data points spread across the lower half of the plot. This suggests that though the case may be far from a target, it is likely to also deliver good results. Importantly, note the higher concentration and larger bubbles in its upper-right quadrant. While in Fig. 9.28, the profile suggest the case has a substantially

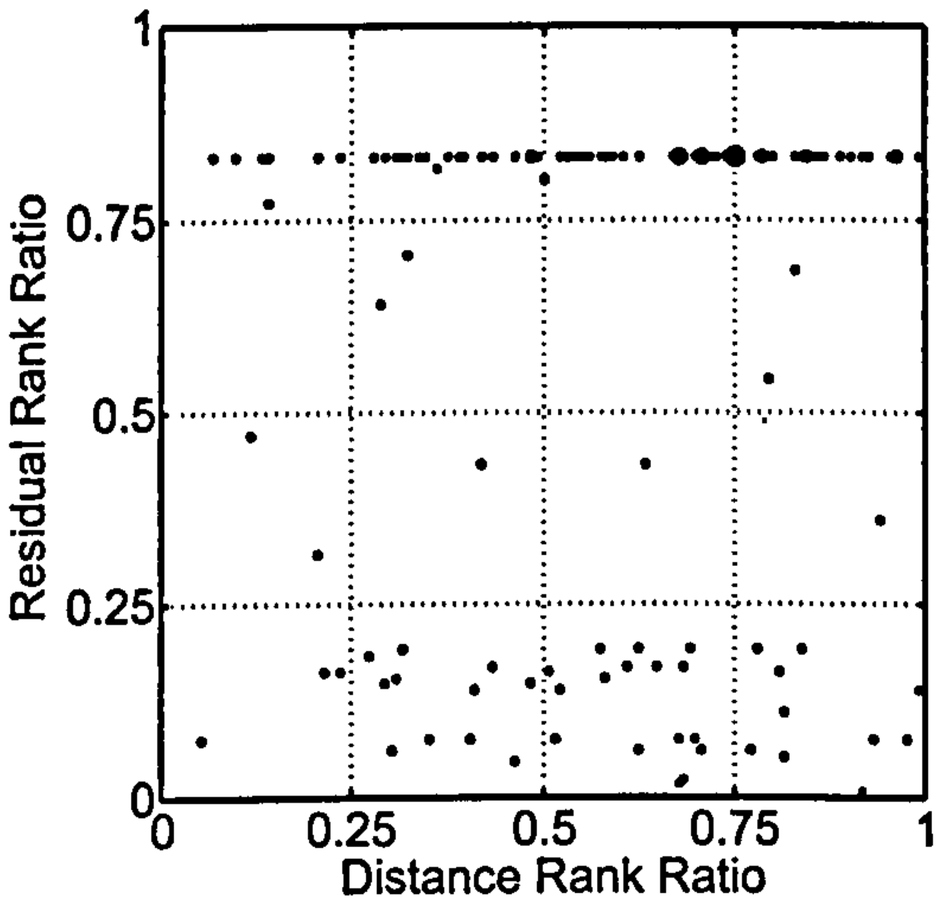

Figure 9.25: Case Profile for Reliable `Case 110` from `Finnish Sample 19`

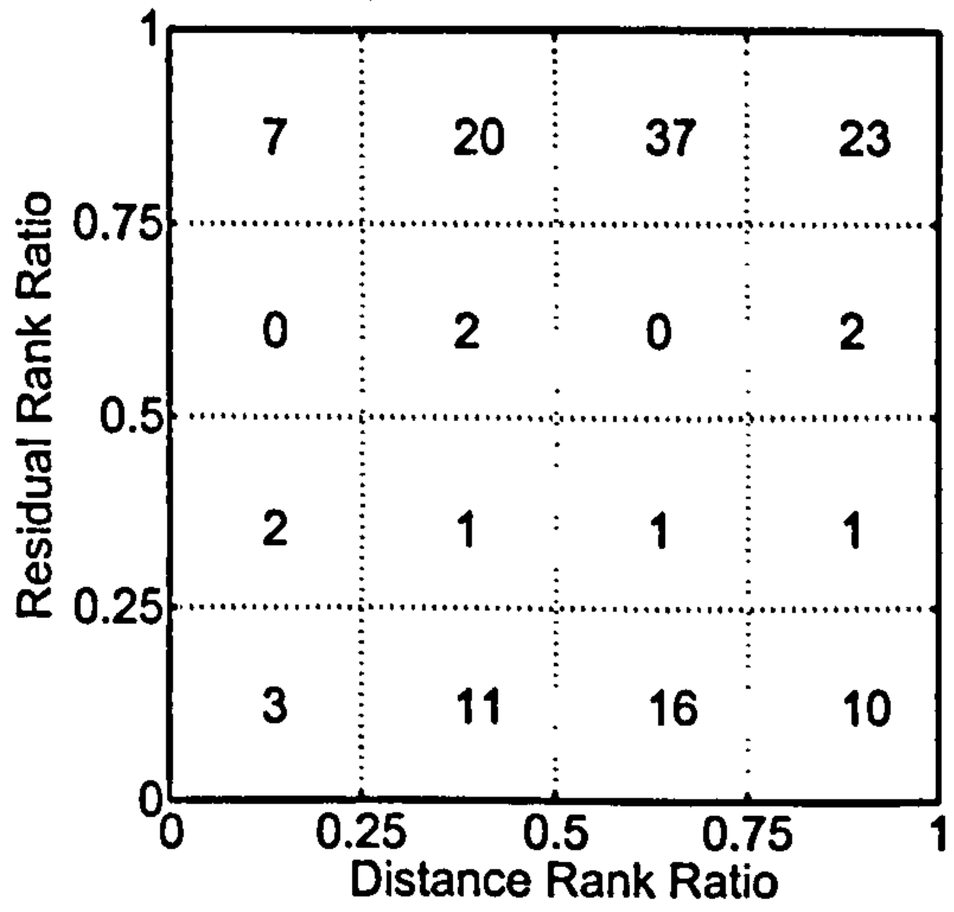

Figure 9.26: Case Profile for Reliable `Case 110` from `Finnish Sample 19`

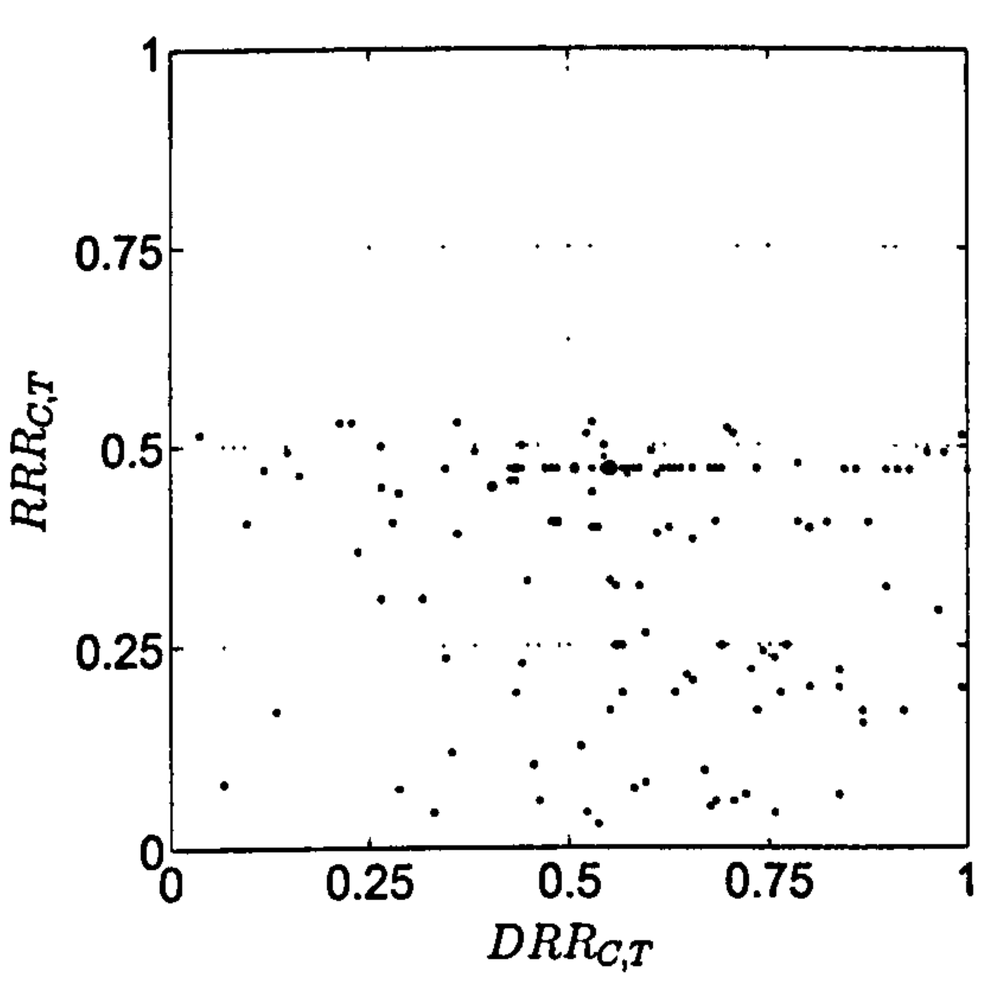

Figure 9.27: Case Profile for Unreliable `Case 77` from `Finnish Sample 20`

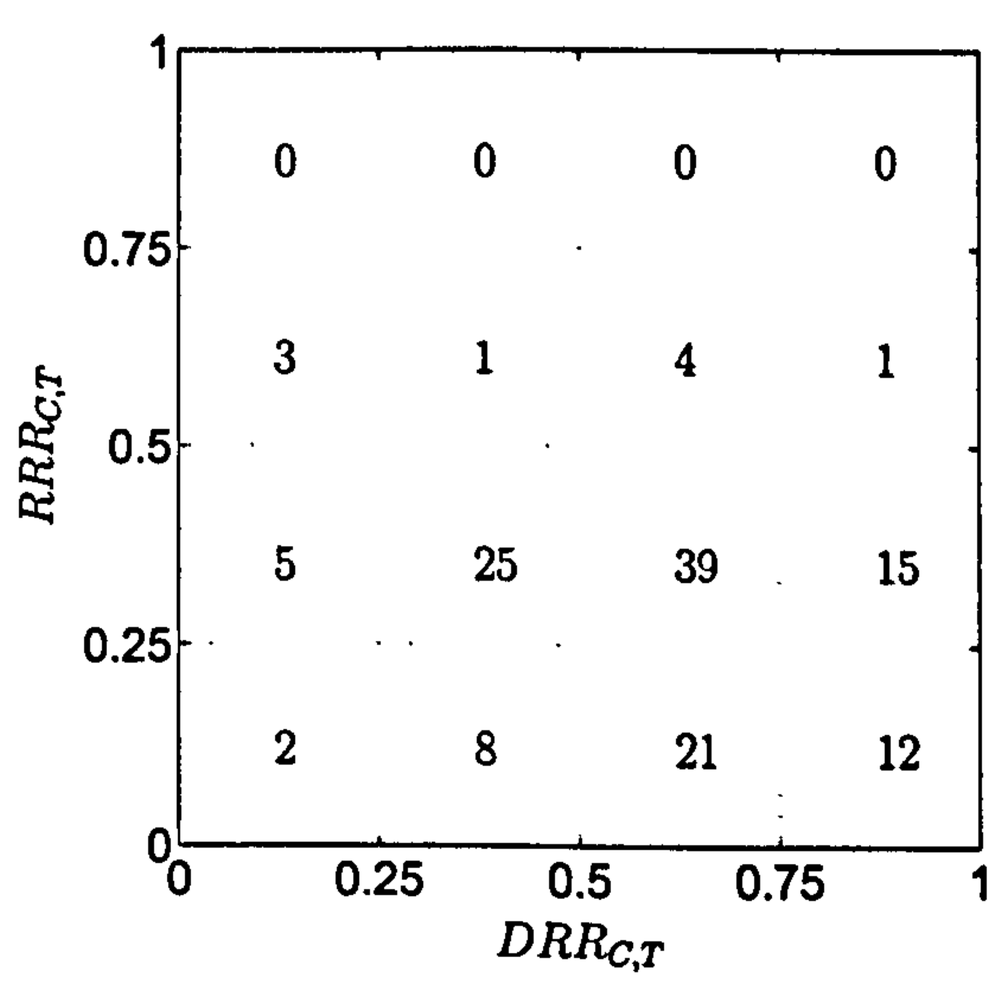

Figure 9.28: Case Profile for Unreliable `Case 77` from `Finnish Sample 20`

different solution value in comparison to other cases in the case base, irrespective of its distance from the target. Considering the consistency of poor results ($RRR_{C,T}$ always is equal to one), such a case must be dealt with severe caution since it seems certain that it usage will deliver substantially poor results again.

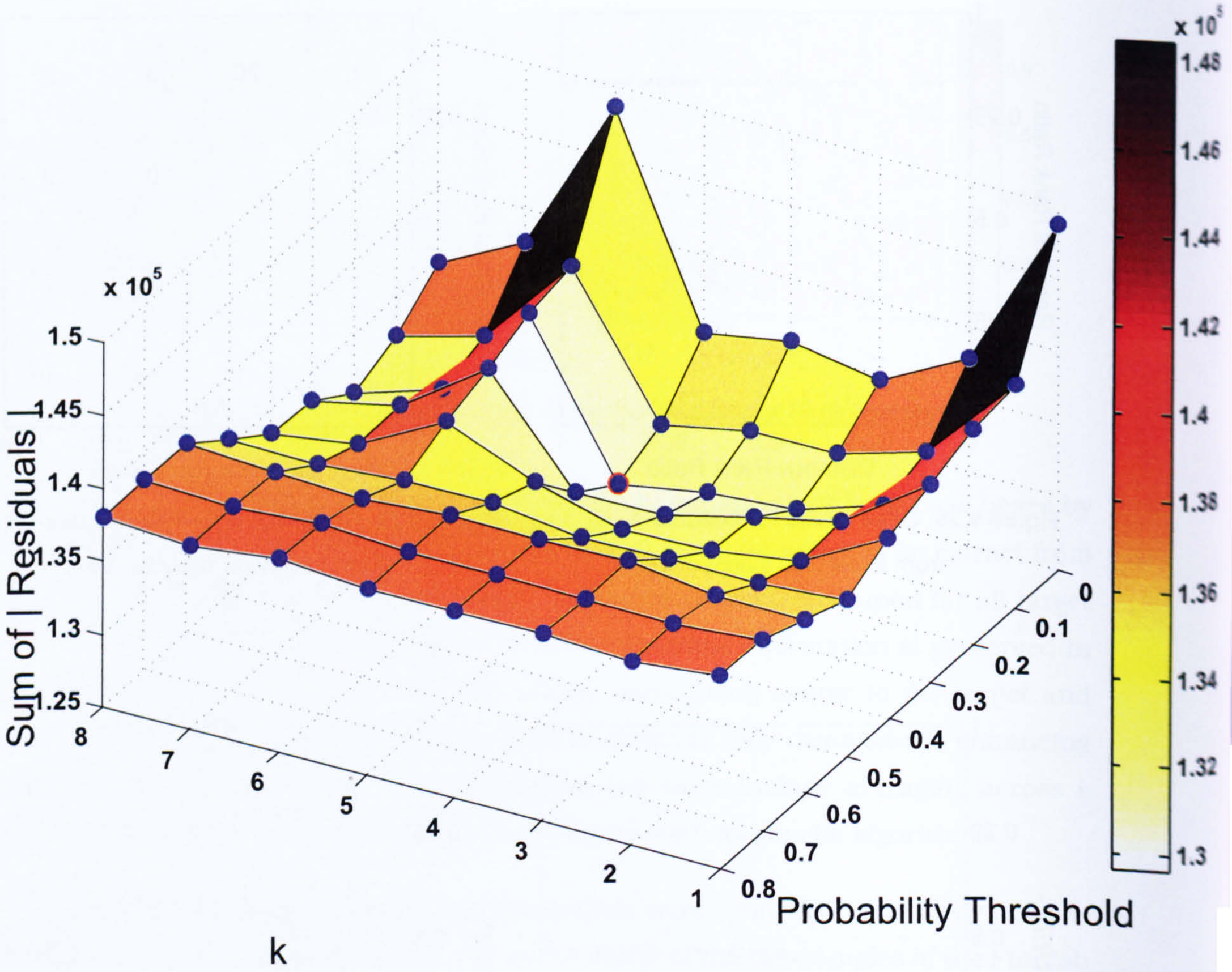

Figure 9.29: Comparison of Performance (Means) by Coupling k-NN and Probability Threshold for the Finnish Dataset

9.3.4 Prediction

Lastly, the impact of using the case profiles on prediction accuracy is examined. Once again, predictions by using P_2 to measure case quality failed. This certainly is because of the degree of unreliability of the cases in the case base. In the case of the Finnish data sets, an early experiment revealed that the optimum combination of k and PT that minimises $Sum|Res|$ was 5 and 0.2 respectively. To check whether the residuals decrease further as k increases, the range of k was increased to $[1 \ldots 8]$. The results from the predictions in the form of means of $Sum|Res|$ for the Finnish data set are reported in Table 9.12 plotted in Fig. 9.29.

Table 9.12: Mean of $Sum|Res|$ from 30 Random Samples of the Finnish Data Set

$\frac{Probability\rightarrow}{k\downarrow}$	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8
1	148576	140213	139672	138436	137407	135871	137057	138259	138402
2	137713	133895	132888	134358	134312	135421	137196	137750	137666
3	134685	132232	130957	132996	133366	135495	137952	137845	138061
4	135713	132157	130488	131638	133241	135206	137710	137906	137973
5	134623	130876	129440	131580	134783	136063	137786	137879	137925
6	148576	140213	139672	138436	137407	135871	137250	138259	138402
7	137713	133895	132888	134358	134312	135421	137419	137750	137666
8	134685	132232	130957	132996	133366	135495	137841	137845	138061

Table 9.13: Kruskal-Wallis Test Results for Finnish Data Set

k	PT	p-value
1	0.5	0.65
2	0.2	0.23
3	0.2	0.09
4	0.2	< 0.05
5	0.2	< 0.05

Again, one observes that using the nearest neighbour provides the highest $Sum|Res|$ or least prediction accuracy. But as the value of k increases, estimates tend to get better as indicated by the decreasing mean of $Sum|Res|$. The mean was recorded lowest for $k = 5$ with an improvement of 9% in prediction accuracy in comparison to $k = 1$. This again may be a result of negation of extreme values by averaging more and more solutions. But as soon as case quality is reflected upon before reuse by setting PT as low as 0.1, we observe a drop in $Sum|Res|$ for all k. This trend only continued until $PT = 0.2$ after which the $Sum|Res|$ began increasing for most values of k since more distant cases were being retrieved. This may be due to few good quality cases in the case bases which may be frequently used despite being distant from the target in the problem space. Even for the larger range of k, the optimum combination of k and PT for this data set was 5 and 0.2 respectively, having the lowest mean of $Sum|Res|$. If one concentrates on $k = 1$, it can observe that as PT increase, the mean of $Sum|Res|$ constantly declined until $PT = 0.5$. This again is a confirmation that the system ably rejected use of unreliable cases in order to increase prediction accuracy. Of course, as k increased, more distant cases must have been used due to discrimination that resulted in increase in sum of residuals.

Lastly, to statistically verify the value of use of the case discrimination system on the Finnish data set, Table 9.13 lists the results from the Kruskal-Wallis tests on the residuals. Similar to Table 9.8, the first and second column record the value of k and the corresponding value of PT which results in the lowest $Sum|Res|$. The third column records the p-value indicating whether the residuals from using the proposed case discrimination system were significantly different to simply using k nearest neighbours at $\alpha = 5\%$. In contrast to the Desharnais data set, the null hypothesis could not be rejected for $k = 1, 2, 3$ in the Finnish data set. This is rather surprising, especially

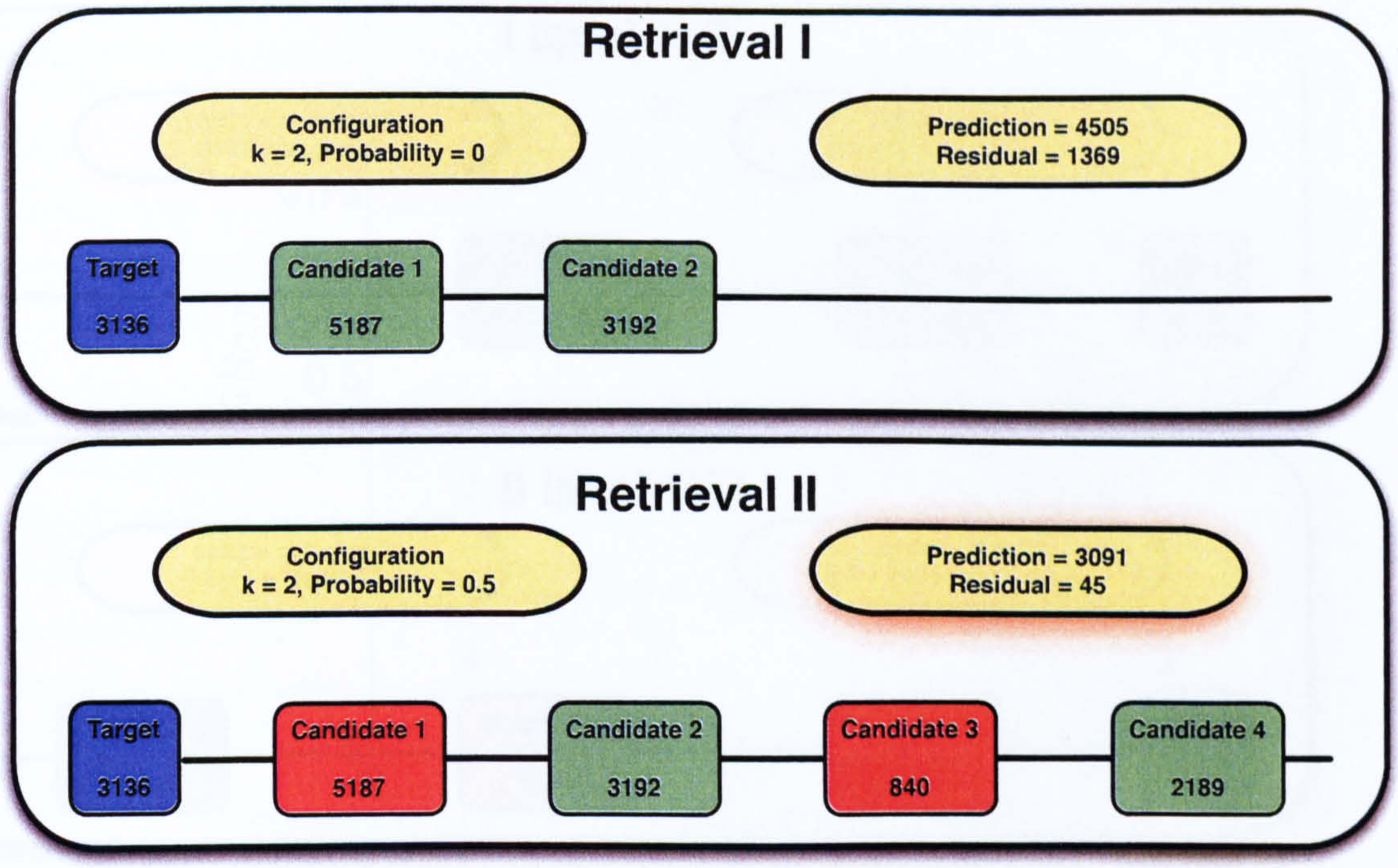

Figure 9.30: An Example of Increase in Prediction Accuracy

for $k = 1$ considering the fall in $Sum|Res|$. However, it is evident from the $Sum|Res|$in all three cases there seems to be at least some value in using the proposed technique. While for remaining values of k, the results are statistically significant and the null hypothesis could be rejected. Hence, the application of the case discrimination system did result in improvement in accuracy, especially for $k = 5$ which also resulted in the lowest $Sum|Res|$.

9.4 Retrieval Demonstrations

The above sections have overall shown that the proposed methods improve prediction accuracy. But the results have so far been looked at from a very broad perspective considering the $Sum|Res|$ were used as a measure of accuracy. In this section, we provide two concrete examples (real retrieval instances) that demonstrate the working of the proposed case discrimination system.

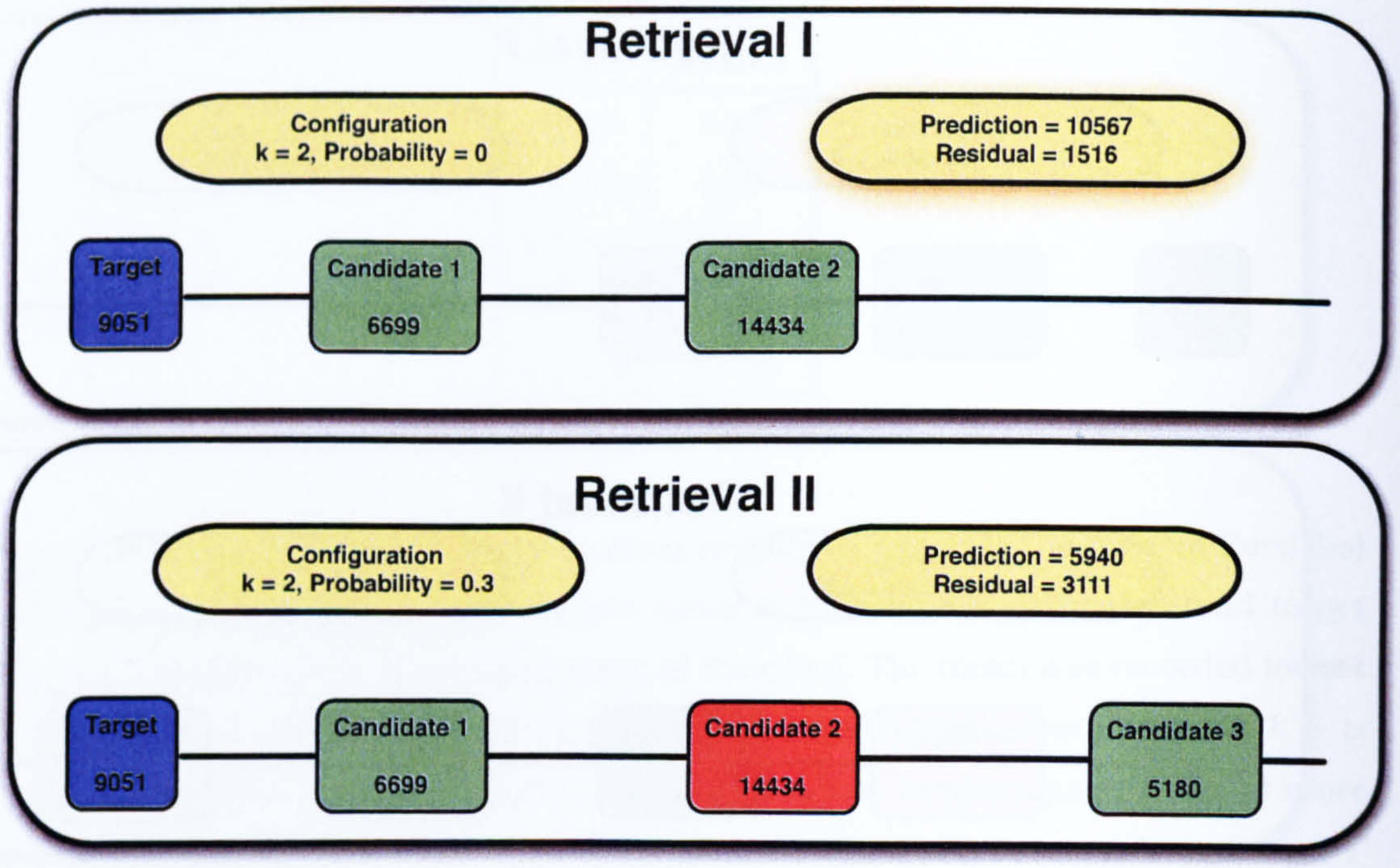

Figure 9.31: An Example of Decrease in Prediction Accuracy

Fig. 9.30 is a retrieval instance from a sample of the Desharnais data set. In the upper panel (Retrieval I), prediction using only nearest neighbours ($k = 2$) is depicted for the target case (in blue). Candidate cases 1 and 2 are the two closest to the target and their solutions are averaged to prediction effort of 4505 hours for the target, leaving a residual of 1369 hours. While in Retrieval II, for the same target case, but with $k = 2$ and $PT = 0.5$, candidates 1 and 3 were rejected because of being unreliable. Instead, candidates 2 and 4 were reused which reduced the residual to only 45 hours (i.e. a 96% increase in accuracy). Note that the distances between the candidate boxes only reflect the order of their distance from the target and not scaled Euclidean distance.

On the contrary, Fig 9.31 is an example where prediction accuracy decreases due because of the case discrimination system. In panel Retrieval I, using $k = 2$ Candidate cases 1 and 2 were used to prediction for the target case with solution value 9051. The residual in this case was 1516 hours. However, as shown in panel Retrieval II, the case discrimination system rejected candidate case 2 and reused 3 instead causing the residual in this case to increase to 3111 hours. The rejection of candidate 2 can be

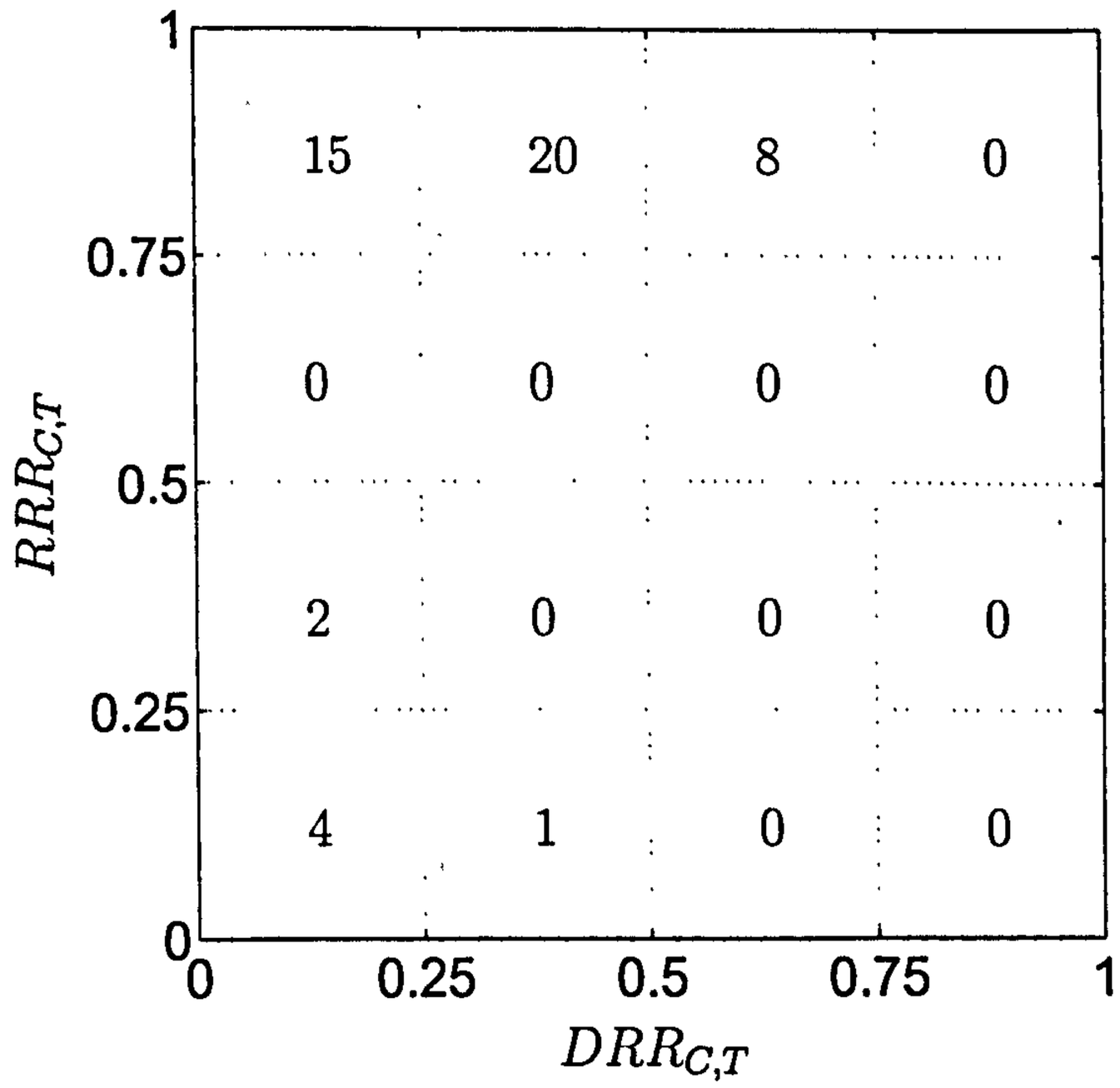

Figure 9.32: Case Profile of Rejected Case

attributed to its large solution value since it may stand out in the solution distribution and resultantly, must have a poor case profile. To confirm this, the relevant case's profile was retrieved and is plotted as Fig. 9.32. Clearly, the case has performed poorly in the case considering the large frequency in the upper-left quadrants of it's profile. Hence there is ample evidence and reason to deem the case unreliable and not use it.

There are several such examples of good and bad retrievals that can be cited. But the underlying message is that there is no assurance that the system may not reject a circumstantially good case.

9.5 Discussion

Having administered the techniques to all three data sets, this section revisits the results to draw a big picture of their effectiveness. To recall the order of techniques implemented, first the Mantel test was performed on each of the 30 random samples

Table 9.14: Summary of Effectiveness of Techniques

	n	Mantel's Test	Visualisations	Enhanced Prediction
BT	18(12)	Y	Y	N
Desharnais	77(51)	Y	?	Y
Finnish	207(137)	Y	N	?

of all three data sets. Thereafter, several visualisations were presented to demonstrate the quality or inherent regularity of the case bases, followed by examination of individual cases. Lastly, the generated case specific information was used to influence case selection for reuse in an attempt to increase accuracy of solutions.

Table 9.14 summarises the efficacy of each technique, i.e. whether it satisfactorily demonstrated case base dissonance and enhanced prediction accuracy, when administered to the three data sets. The first column records the total number of cases used while the number within brackets indicate the size of the case base. The symbols 'Y' and 'N' indicate whether the respective techniques were effective or not respectively, while '?' indicates ambiguity about effectiveness of the technique.

The Mantel test proved to be very valuable to indicate the quality of all samples (or case bases) being dealt with for each data set. Higher values of correlation were observed for the BT data set suggesting more inherent problem-solution regularity in the case case. While the correlation values were lower for Desharnais and Finnish data sets suggesting the presence of more irregularity. The Mantel test also helped reflect upon the applicability of the case bases for CBP via the significance test. This supports the notion of a true relationship between the predictor and response variables in the real world, which helps explain causality.

The visualisations more explicitly manifested the quality of the samples of each data set examined. However, they seem useful for smaller data sets (BT) since it is possible to clearly view overall dissonance and identify suspect unreliable cases. But this approach seems less pragmatic as the case base size increases since the plots appear cluttered making it nearly impossible to manually extrapolate any meaningful trends and search for unreliable cases. Hence, this is a major constraint since case bases are generally large (i.e. at least > 50) for most domains making the plots more or less unhelpful. However, individual case visualisations, such as in Figs. 9.14 and 9.14

were more interesting as their respective behaviour and role in the case base could be studied.

Lastly, the distance metric was supplemented with case quality information to selectively use high quality or reliable cases for prediction. Overall, it found that applying the technique did positively influence solution accuracy. But the degree of impact was dependent upon the regularity of the case base itself. In a fairly regular case base (BT), the technique seemed to make a modest impact on prediction accuracy since it may have had fewer opportunities to reject unreliable similar cases. However, in the case of the larger and more irregular case bases, the advantages of using the proposed techniques were more obvious considering the reduction in sum of residuals and results from statistical tests.

Thus overall, this chapter shows some value in the proposed techniques for prediction problems although the degree of impact may be variable. The results clearly show that it may not always be valuable to use the most similar case for reuse without considering its performance history or reliability. Users have to be even more cautious when reusing simply the nearest neighbours or statistical adaptation techniques such as distance-weighted average that give more importance to nearer cases. The conclusions of this research follow in the next chapter which revisits the aims and objectives and assess the degree of which they have been attained. Thereafter, directions for further maturing this work have been put forth.

IV

Conclusions

PAGE

MISSING

IN

ORIGINAL

Chapter 10

Conclusions

This chapter reviews the research presented in this thesis and reflects upon the work's significance and limitations. To begin with, the research objectives (discussed in Chapter 1) are revisited and assessed to determine the degree to which they have been met. Thereafter, the contributions made by this research are highlighted followed by current restraints of the presented work. Lastly, potential directions for further work are explored.

10.1 Summary of Research

A study of the application of Case-Based Prediction (CBP) to software engineering estimation was conducted which shed light on the criticality of a competent case base for delivery of effective solutions. It was expected that an impression or measure of inherent regularity in the case base would influence users' confidence, or caution them with regard to the delivered solution's reliability. A review of current literature in CBR showed that methods to assess case base quality existed, though they largely catered for analytic problem domains e.g. classification and diagnosis. These techniques were incapable of accommodating the extent or unboundedness of problem and solution spaces and resulting complexity of problem domains.

Thus, the need to research alternative techniques for assessment regularity of case bases for CBP arose. A novel technique to measure the overall regularity of the case base was experimented with. Thereafter, two visualisation techniques that highlighted the regularity of the examined case base were demonstrated. These visualisations also enabled identification of unreliable cases (subject to practical limitations) that

appeared to be the cause of overall distortion in the case base's regularity. Additionally, an objective method to measure individual case reliability was introduced that strengthened the notions of individual case reliability drawn from the visualisations.

All of the above was undertaken keeping the larger goal of enhancing solution accuracy in mind. The idea was to identify unreliable cases and confine their usage to extenuate their effect on solution quality. The proposed techniques were demonstrated upon three software engineering data sets of different sizes, but it was observed that their effect varied across from one data set to another.

10.2 Research Objectives

The four objectives mentioned in Chapter 1 will now be restated to be assessed in light of the work presented in the preceding chapters.

- **OB1:** *To establish that there is a need for an alternative technique that addresses problem–solution irregularity in CBP domains.*

  In Chapter 4, many recent approaches to case base maintenance were revisited. It was shown that these techniques were developed for analytic problems and were unsuitable to be applied for prediction problem domains. This is a result of the continuous value solutions in CBP that makes the proposed approaches ineffective since many concepts such as case coverage and reachability cannot be defined with certainty. Hence, there was a need to develop new approaches that addressed the challenges posed by prediction problem domains.

- **OB2:** *To assess the problem-solution regularity in a case base to determine its applicability to CBP and influence user confidence.*

  A case base, in which cases are proportionally placed with respect to each other in both problem and solution spaces, is more likely to deliver a good quality solution in comparison to other case bases where cases may be disproportionably placed. While a regular case base is desirable, case bases in the real world tend to exhibit some distortions. Hence, it is valuable to have an impression of the degree of distortion to verify its applicability to CBP and influence confidence on the proposed solution.

This research successfully used the Mantel's randomisation test to measure regularity of case bases from prediction problem domains. The results showed that different samples of the case bases exhibited varying degrees of regularity between the problem and solution spaces. It made possible to identify relatively more regular case bases that would gain more user confidence since they are likely to deliver good solutions. Additionally, varying degrees of regularities amongst different samples indicated the presence of noisy or unreliable cases in the case base.

- **OB3:** *To identify unreliable cases that distort the overall regularity of a case base.*

While the Mantel's randomisation test indicated the presence of unreliable cases, it is vital that such cases are identified to either be treated or avoided from reuse. To achieve this, some novel visualisations were proposed that made it possible to identify cases that exhibited considerable irregularity and could potentially be labelled as unreliable. But this was more challenging and inefficient for larger case bases since the visuals appeared more cluttered and hence, it was difficult to manually identify unreliable cases. Besides the problem of dealing with large case bases, the visualisation also delivered a subjective impression of case reliability. As an alternative, Spearman's rank correlation was used to provide a measure of case reliability and in addition, it also confirmed results from the visualisations.

- **OB4:** *To investigate if case reliability can gainfully supplement inter-case distance measures to increase solution accuracy in CBP using the domain of software engineering estimation as an example.*

A case discrimination system was proposed that successfully avoided reuse of unreliable cases in favour of distant yet reliable cases. The reliability information was gathered from case profiles that tracked the performance of the candidate case. Although positive, the impact on accuracy varied from one case base to the other depending upon the regularity. In regular case bases (BT), the impact was modest considering reuse of few unreliable cases must have been avoided while the increase in accuracy was marked for larger and more irregular case bases (Desharnais and Finnish).

10.3 Summary of Research Findings

The major finding from this research is that:

> It is beneficial to supplement inter-case distance with a reflection of the candidate case's quality and reusing relatively distant yet reliable cases for prediction increases solution accuracy i.e. using the nearest candidate case(s) for prediction is not always desirable.

Other important findings include:

- *Software engineering data sets maybe vulnerable to irregularity and should be treated cautiously.* It was observed that in two of the three data sets used in this research (Desharnais and Finnish), there was substantial irregularity in the case bases. In the remaining data set (BT data set), the irregularity was insignificant but this could be a result of its small size as reflected by the differences in the samples. Overall, this suggests that SE data sets are likely to contain noise and therefore, each case should be carefully examined before use. The degree of impact from irregularity may differ from one technique to another built using the data set. For example, a single outlier could significantly alter a regression model and resultantly decrease overall prediction accuracy. On the other hand, the presence of a single outlier in CBP would have a rather modest impact considering the likelihood of it being frequently reused. Variation of such regularity is one such plausible explanation for erratic prediction accuracy.
- *Different optimum values of k for different data sets.* Another interesting observation drawn from results is that while there is a consensus on using more than one nearest neighbour for prediction, the optimum value of k differs from one data set to another. This may again be an artefact of the irregularity in the case base. More regular case bases are likely to perform well using the nearest neighbour. On the other hand, irregular case bases would benefit from using $k > 1$ since they may neutralise extreme values.
- *Measuring case reliability is always favourable.* It was noted that using a case discrimination system that learns to overlook unreliable cases always increases prediction accuracy (except for $k = 5$ in the Finnish data set). This is true even at the lowest experimented threshold value (0.1). Hence, one is better off using

any such system along with the similarity measure irrespective of the value for k. However, the degree of impact is determined by the degree of irregularity in the case base. It is expected that the improvement in prediction may be rather modest for smaller and regular cases while it may be more significant for larger and irregular case bases. The increase in impact for irregular case bases is because there are higher chances and instances of ignoring unreliable cases.

- *Influence on user confidence.* The reason why prediction accuracy increases when using the proposed case discrimination system is because cases that are more regularly and coherently spread out in the problem and solution spaces are used to predict each other. This is contrary to using only k where prediction accuracy increases with increasing k mostly due to extreme values negating each other. With such explanatory knowledge, it is likely to increase the confidence of users on the system and solutions delivered.

10.4 Contributions of the Thesis

The following contributions have been made by this work to the field of Case-Based Prediction:

I — A gap in contemporary research pertaining to maintenance in CBR has been brought to light. It has been shown that existing methods and techniques for case base maintenance are unfit for application to CBP.

II — A technique novel to CBR has been demonstrated to measure problem-solution regularity in case bases used for prediction.

III — Additionally, two new visualisations have been presented to gauge inherent distortion of regularity in case bases and individual cases. Observations have been further substantiated by an objective measure of individual case reliability.

IV — A novel concept of case profile has been introduced that records the performance history of a case to enable judging its reliability for future reuse.

V — Lastly, a relatively simple yet effective approach has been developed to enhance solution accuracy that supplements inter-case distance with candidate case quality information to selectively reuse similar cases for prediction.

10.5 Limitations of Work

While the proposed techniques have been demonstrated to be useful, there are some limitations of the research that are recognised and are discussed in this section.

10.5.1 Limitations of Approach

- All the visualisations of different case bases proved to be effective in demonstrating the inherent dissonance. However, they were relatively unhelpful for large case bases when trying to manually identify suspect unreliable cases. This was shown clearly by sections 9.2.2 and 9.3.2, especially the latter. Thus, the visualisations fell short of their purpose and utility for practical reasons.
- The research used the $Sum|Res|$ as an indicator for error. Though this is widely accepted to be an unbiased measure of error, the technique could have shown to be even more promising by using other measures, especially Pred25. This measure is the count of total number of predictions that were within a range of 25% from the true value. Hence, it indicates the number of projects which have been predicted within a certain margin of error even though the sum of residuals may be higher.
- The frequentist approach used to select cases for reuse has been effective at reducing error. However, it may be regarded as a very simple technique for making such decisions. Another possibility is to use neural networks that can be trained to discriminate against cases that appear to have unreliable profiles. Likewise, other techniques capable of learning patterns can be explored for use that may perhaps lead to better results.
- The size of the case base has been kept constant during the experiments. This does not adequately represent a real world scenario where cases are at least aperiodically added to the case base.

10.5.2 Tool Limitations

- The implementation of the work presented was coded in MATLAB. Though it is possible to create stand-alone executable applications using its in-built compiler,

currently this has not been done. Hence, it is possible to share only the scripts within the community to widen usage and validation, but this requires users to have access to MATLAB which is a relatively expensive piece of software.

- Secondly, the scripts were coded specifically keeping the research objectives in mind. Hence, they lack flexibility and other desirable features such as choice of distance metrics, feature and case subset selection etc. Additionally, at present there also exists no GUI to ease usability.

Though these tool limitations are constraining, they do not pose a difficult hurdle for further advancement of the work. This is because the details of the technique have been provided in our methodology chapters and can be easily built into systems by developers. Other alternatives have been presented in Section 10.6.1

10.5.3 Analysis Limitations

So far, the technique has been shown to be effective using a manually filtered set of case base features rather than a pre-determined optimal subset using wrappers [KS02a, AB94]. It would have been interesting to determine the effect of the presented technique in conjunction with feature selection i.e. verify if there is further improvement in prediction accuracy when already using an optimal set of features for measuring similarity. Another worthwhile comparison of performance would be between the presented technique and case subset selection. Since, the discrimination system implicitly deletes (or ignores) cases at different levels of PT, it conceptually strives to achieve the same results as case subset selection in which an optimal set of cases are selected to comprise the case base. Besides performance accuracy, other parameters that the two techniques could be compared against include computational expense, reliability and suchlike.

Secondly, the experiments have been conducted keeping the case base size constant. But realistically, cases are added to the case base from time to time and the system must gauge the reliability of each new case. This can be simulated for research purposes by adding every test case to the case base once a prediction has been made. Upon addition, a case profile has to be populated for the newly added case in the same manner as previously for the other existing cases. Thus, such experimentation in a

more realist environment would prospectively give us deeper insight into the long-term effectiveness of the technique.

Besides, for more robust validation, it is imperative that the technique is used on more software engineering data sets. This would further support the value of the technique when predicting software costs. In addition, the technique could also be tested upon data sets from other problem domains.

10.6 Future Extensions to Research

Suggestions for future work to build upon current research are made in this section. Three different areas of work are discussed in the following sections.

10.6.1 Tool Support

The current implementation of this work was coded using scripts in MATLAB [MAT] and lacks a GUI. Hence, it is essential that these scripts are upgraded with a user-friendly GUI to make them available for widespread use and validation within the CBR community.

A rather preferable alternative would be to upgrade ArchANGEL [KSH02] which is a CBP shell developed previously by our research group. ArchANGEL, which is freely available, has been extensively used by researchers worldwide for conducting activities pertaining to software engineering estimation. Embedding the technique into ArchANGEL would augment the tool's utility and further allow more opportunities for validation of our technique by numerous users.

Lastly, an exciting possibility would be to contribute the technique as a method(s) (or class(es) in Java) in jColibri [BTGCDA04] which is a very flexible CBR shell. The shell is designed to enable handling multitudes of data types and CBR pertaining operations by developing new or modifying existing methods. Importantly, the tool appears to be gaining widespread popularity within the CBR community and hence, it shall provide a remarkable platform to validate our techniques across a variety of domains by different users.

10.6.2 Method Refinement and Enhancement

The populated case profiles are currently static. As a result, once the profiles are generated, unreliable cases are as good as deleted for various values of PT since they will never be used. Instead, *updated beliefs* can be used. Here, even though the prediction for a test case is made using the k nearest neighbours, a check is performed on the quality of solution delivered if every case in the case base was reused exclusively. This would result in more meta-data for each prediction and accordingly, each candidate case's profile can be updated. In this way, cases which were previously considered unreliable may gradually begin to be reused and deliver effective solutions. The utility of this approach could be better understood with an example. Imagine a scenario where a software project (existing in the case base) was completed using a revolutionary technique that substantially improved productivity. Hence, in relation to other similar candidate cases, this case may be initially deemed unreliable due to its differences with other cases in the solution space. However, over time as this new technology becomes popular, the usefulness of this unreliable case would be realised by its prospects of providing a good quality solution and henceforth, be favoured for reuse over other candidate cases.

Secondly, it is obvious that the technique that decides whether to use a case or not is crucial for meeting our objective of increasing prediction accuracy. It is recognised that the frequentist approach used in this research to measure quality though effective, is relatively simple and naïve. This leaves considerable room to test alternative methods that could be used instead such as fuzzy logic, neural networks, decision trees, logistic regression, support vector machines and so on.

Lastly, the case profiles can be investigated in more depth and could be used to classify new cases as reliable, unreliable, common, distinct etc. This can be achieved by examining their distinct patterns or spread of data points in their respective profiles that reveal characteristics of the case, such as in Figs. **??** and **??**. Such classifications can be further used to influence other decisions such as whether to add a case to the case base, measure expected solution quality or even measure expected frequency of use.

10.6.3 General Applicability

Most importantly, there is a need to validate the general applicability of the techniques presented in this research. This may initially be done strictly on other CBP systems to uncover any unspotted weaknesses in the system that need to be addressed. Thereafter, the technique can be further modified to be factored into other CBR systems e.g. classification tasks, design and planning tasks, recommender systems, etc.

10.6.4 Software Engineering Estimation

Notwithstanding the above, this research also raises questions about software engineering data sets. It has been shown that there is a need to reflect upon data collection practices including both what and how data is collected. While the latter issue is more administrative in nature, the former underlines the necessity of more research in the field to further understand the dynamics of software development. With more insight, it may also open channels to develop adaptation routines that could perhaps be beneficial to reduce or even eradicate problems caused by heterogeneity of software engineering data sets.

10.7 Summary

This research has shed light upon the fact that CBP systems used for software engineering estimation are sensitive to the presence of noise in the case bases. And this may very well be true for case bases from other prediction domains which are vulnerable to similar irregularities. Since this may have an adverse effect on accuracy of solutions, it is vital that methods are in place that monitor the quality of case bases and check for unreliable cases to avoid their reuse. Of course, this is true for other CBR systems too that rely upon problem-solution regularity within their case bases to perform well. While this research has presented an alternative to account for unreliable cases in prediction domain case bases, the techniques deployed for such purposes may differ from one application to the other and hence, need to be appropriately chosen or developed.

V

Appendices

PAGE

MISSING

IN

ORIGINAL

```
fopen_sumerrors = fopen((strcat('BT',num2str(iterations), ...
        '/SumOfResiduals_',date,'.txt')), 'a');


fprintf(f_open, '*****************************************' ...
   '*********************************\n\n');
fprintf(f_open, 'Date of Run: %s\t\t', date);
fprintf(f_open, 'Time: %d:%d:%d\n\n', clock_(4), clock_(5), clock_(6));
fprintf(f_open, '*****************************************' ...
   '*********************************\n\n');

fprintf(fopen_sumerrors, '*****************************************'...
   '*********************************\n\n');
fprintf(fopen_sumerrors, 'Date of Run: %s\t\t', date);
fprintf(fopen_sumerrors, 'Time: %d:%d:%d\n\n', clock_(4), clock_(5), ...
   clock_(6));
fprintf(fopen_sumerrors, '*****************************************'...
   '*********************************\n\n');


colnames = fieldnames(data);

[dataset_train, dataset_test] = split_data(data, 1,0);

size_train = size(dataset_train);
size_test = size(dataset_test);

cb_profile = zeros(size_train(1), 16);
metadata = zeros((size_train(1) * (size_train(1)-1)),6);
metadata_counter = 1;
metadata_counter_ = 1;

dist_matrix = zeros(size_train(1));
resi_matrix = zeros(size_train(1));
pred_residuals = zeros(5,9,size_test(1));
```

```
% Preparing data for similarity matching

for target = 1:size_train(1)

   x = 1;

for candidate = 1:size_train(1)

   if (target == candidate)
      continue;

   else

      %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
      %         Normalise Continuous Features
      %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

      ranges = normalise(dataset_train, dataset_train(target));

      distance = calc_distance(dataset_train(target), ...
         dataset_train(candidate), ranges);

      distances(x,1) = candidate;
      distances(x,2) = distance;

      residual = dataset_train(target).(colnames{1})-...
      dataset_train(candidate).(colnames{1});

      residuals(x,1) = candidate;
      residuals(x,2) = residual;

      metadata(metadata_counter, 1) = target;
      metadata(metadata_counter, 2) = candidate;
      metadata(metadata_counter, 3) = distance;
      metadata(metadata_counter, 5) = residual;
```

```
        metadata_counter = metadata_counter + 1;
        x = x + 1;

    end
    end

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Sorting distances and residuals of all Candidate cases
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


    distances = sortrows(distances, [2]);
    residuals = sortrows(abs(residuals),[2]);

    for candidate = 1:size_train(1)

        pack;

        if (target == candidate)

           dist_matrix(target, candidate) = 0;
           resi_matrix(target, candidate) = 0;
           continue;

        else

           for rank_seek = 1:(size_train(1)-1)
              if (distances(rank_seek,1) == candidate)
                 metadata(metadata_counter_,4) = rank_seek / ...
                 (size_train(1)-1);
              end
           end
```

```
for rank_seek = 1:(size_train(1)-1)
   if (residuals(rank_seek,1) == candidate)
      metadata(metadata_counter_,6) = rank_seek / ...
      (size_train(1)-1);
   end

   dist_matrix(candidate, target) = metadata(metadata_counter_,4);
   % Distance Matrix for Mantel's Test

   resi_matrix(candidate, target) = metadata(metadata_counter_,6);
   % Residual Matrin for Mantel's Test

end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%         Populating Case Profile
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if metadata(metadata_counter_,4) <= 0.25

   if metadata(metadata_counter_,6) <= 0.25
      cb_profile(candidate,1) = cb_profile(candidate,1)+1;
   elseif metadata(metadata_counter_,6) <= 0.5
      cb_profile(candidate,5) = cb_profile(candidate,5)+1;
   elseif metadata(metadata_counter_,6) <= 0.75
      cb_profile(candidate,9) = cb_profile(candidate,9)+1;
   elseif metadata(metadata_counter_,6) <= 1
      cb_profile(candidate,13) = cb_profile(candidate,13)+1;
   end

elseif metadata(metadata_counter_,4) <= 0.5

   if metadata(metadata_counter_,6) <= 0.25
```

```
            cb_profile(candidate,2) = cb_profile(candidate,2)+1;
        elseif metadata(metadata_counter_,6) <= 0.5
            cb_profile(candidate,6) = cb_profile(candidate,6)+1;
        elseif metadata(metadata_counter_,6) <= 0.75
            cb_profile(candidate,10) = cb_profile(candidate,10)+1;
        elseif metadata(metadata_counter_,6) <= 1
            cb_profile(candidate,14) = cb_profile(candidate,14)+1;
        end

    elseif metadata(metadata_counter_,4) <= 0.75

        if metadata(metadata_counter_,6) <= 0.25
            cb_profile(candidate,3) = cb_profile(candidate,3)+1;
        elseif metadata(metadata_counter_,6) <= 0.5
            cb_profile(candidate,7) = cb_profile(candidate,7)+1;
        elseif metadata(metadata_counter_,6) <= 0.75
            cb_profile(candidate,11) = cb_profile(candidate,11)+1;
        elseif metadata(metadata_counter_,6) <= 1
            cb_profile(candidate,15) = cb_profile(candidate,15)+1;
        end

    elseif metadata(metadata_counter_,4) <= 1

        if metadata(metadata_counter_,6) <= 0.25
            cb_profile(candidate,4) = cb_profile(candidate,4)+1;
        elseif metadata(metadata_counter_,6) <= 0.5
            cb_profile(candidate,8) = cb_profile(candidate,8)+1;
        elseif metadata(metadata_counter_,6) <= 0.75
            cb_profile(candidate,12) = cb_profile(candidate,12)+1;
        elseif metadata(metadata_counter_,6) <= 1
            cb_profile(candidate,16) = cb_profile(candidate,16)+1;
        end
    end

    metadata_counter_ = metadata_counter_ + 1;
```

```
    end

end

sort_dist_matrix = dist_matrix;
sort_resi_matrix = resi_matrix;

[sort_dist_matrix1, sort_resi_matrix1] = sort_matrices_bydistance ...
(sort_dist_matrix, sort_resi_matrix, size_train(1));

% Spearman's Rank Correlation

case_profile_all = calc_case_profiles(sort_dist_matrix1, ...
   sort_resi_matrix1);

% Re-arranging matrices for case base visualisation

[dist_matrix, resi_matrix] = arrange_corr_matrices(dist_matrix, ...
   resi_matrix, size_train);



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%        Mantel's Randomisation Test
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

mantels_corr(1) = calc_mantels_corr(dist_matrix, resi_matrix, ...
   size_train(1));

max_corr = mantels_corr(1);
min_corr = mantels_corr(1);

for rand_perms = 2:5000
```

```
    rand_dist_matrix = randomise_dist_matrix(dist_matrix);
    mantels_corr(rand_perms) = calc_mantels_corr(rand_dist_matrix, ...
       resi_matrix, size_train(1));

    if mantels_corr(rand_perms) > max_corr
       max_corr = mantels_corr(rand_perms);

    elseif mantels_corr(rand_perms) < min_corr
       min_corr = mantels_corr(rand_perms);

    end

 end

 fprintf(f_open, 'Matrix Correlation\t=\t%d\n', mantels_corr(1));
 fprintf(f_open, 'Max Mantel\t\t\t=\t%d\n', max_corr');
 fprintf(f_open, 'Min Mantel\t\t\t=\t%d\n', min_corr');


 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%***********
 %       Predicting Target Cases
 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

 sum_error_k = 0;
 sum_error_pl = zeros(5,9);


 for target = 1:size_test(1)

    clear distances;
    clear residuals;

    x = 1;
```

```
for candidate = 1:size_train(1)

   ranges = normalise(dataset_train, dataset_test(target));
   distance = calc_distance(dataset_test(target), ...
      dataset_train(candidate), ranges);

   distances(x,1) = candidate;
   distances(x,2) = distance;

   residual = dataset_train(target).(colnames{1})-dataset_train ...
   (candidate).(colnames{1});

   residuals(x,1) = candidate;
   residuals(x,2) = residual;

   x = x + 1;
end

distances = sortrows(distances, [2]);
residuals = sortrows(abs(residuals),[2]);

for dist_rank = 1:size_train(1)
   distances(dist_rank,3) = dist_rank / size_train(1);
end

fprintf(f_open, '\n\nFor Case %d (%d) :\n', target, dataset_test ...
   (target).(colnames{1}));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%***********
%        Nearest k
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for k = 1:5
   [cases_k, solutions_k, prediction_k, error_k] = predict_k ...
   (k, dataset_train, dataset_test(target), distances);
```

```
        for k_counter = 1:k
            fprintf(f_open, 'Case(p2) (%d): %d (%d)\t', k, ...
                cases_k(k_counter), solutions_k(k_counter));
        end
        fprintf(f_open, 'Prediction(k): %d\t Residual(k): %d\n', ...
            prediction_k, error_k);
            pred_residuals(k,1,target) = error_k;
            pred_residuals_(residual_counter,k,1) = error_k;
            sum_error(iterations, k,1) = sum_error(iterations, k,1)+...
            error_k;
        end



        fprintf(f_open, '\n');


        for k = 1:5
            for p = 0.1:0.1:0.8
                [cases_p2, solutions_p2, prediction_p2, error_p2] = ...
                    predict_p (k, p, 2, dataset_train, dataset_test(target),...
                    distances, cb_profile);
                sum_error(iterations, k, ((p * 10)+1)) = sum_error ...
                    (iterations, k, ((p * 10)+1))+error_p2;
                for k_counter = 1:k
                    fprintf(f_open, 'Case(p2) (%d,%s): %d (%d)\t', k,          ...
                        num2str(p), cases_p2(k_counter),...
                        solutions_p2(k_counter));
                end
                fprintf(f_open, 'Prediction(k): %d\t Residual(k): %d\n',      ...
                    prediction_p2, error_p2);
                pred_residuals(k,((p * 10)+1),target) = error_p2;
                pred_residuals_(residual_counter,k,((p * 10)+1)) = error_p2;
            end
        end
```

```
    residual_counter = residual_counter + 1;

end

toc;

    fprintf(f_open, '*************************************************' ...
       '*******************************\n\n');

    fclose(f_open);
    save((strcat('BT',num2str(iterations),'/MATFILE_',date)));
    fclose(fopen_sumerrors);
    keep3 iterations sum_error pred_residuals_ residual_counter;

end
```

A.2 Splitting Data sets into Training and Testing Samples

```
function [dataset_train, dataset_test] = split_data(data, choice, rand)

% function [dataset_train, dataset_test] = split_data(data, choice, rand)
% data = struct data
% choice = split - 1 for 2:1 and 2 for 1:1
% rand = randomise data or not - 1 = yes, 0 =  no

data_copy = data;
size_data = size(data);

even = 0;
if (rem(size_data(1),2))
    even = 0;
else
    even = 1;
end

if (choice == 2)

    train_size = ( round (size_data(1) / 2));
    test_size = size_data(1) - train_size;

elseif (choice == 1 && even ==0)

    train_size = ( round ( size_data(1) / 3) * 2 ) - 1;
    test_size = size_data(1) - train_size;

elseif (choice ==1 && even ==1)

    train_size = ( round ( size_data(1) / 3) * 2 );
    test_size = size_data(1) - train_size;
```

```
end



if (rand == 1)
    rand_num = randperm(size_data(1));
    for i = 1:size_data(1)
        data_copy(i) = data(rand_num(i));
    end
end

dataset_train = data_copy(1:train_size);
dataset_test = data_copy(train_size+1:end);
```

A.3 Normalising Distances

```
function [ranges] = normalise(data, target)

data(end+1) = target;

colnames = fieldnames(data);
size_data = size(data);
size_cols = size(colnames);

ranges = 0;
temp = 0;

for i = 1:(size_cols(1))
    if isnumeric(data(1).(colnames{i}))
        for j = 1:size_data(1)
            temp(j) = data(j).(colnames{i});
        end
```

```
        ranges(i) = max(temp) - min(temp);

        if ranges(i) == 0
            ranges(i) = 1;
        end

    else
        ranges(i) = 0;
    end
end

ranges = ranges';
```

A.4 Calculating Inter-Case Distance

```
function [distance] = calc_distance(data1, data2, ranges)

distance = 0;
colnames = fieldnames(data1);
size_data = size(colnames);

for i = 2 : (size_data(1))
   if isnumeric(data1.(colnames{i}))
      distance = distance + (((data1.(colnames{i})-data2.(colnames{i}))...
         / ranges(i)) ^ 2);
   else
      if (strcmp(data1.(colnames{i}), data2.(colnames{i})) == 0)
         distance = distance + 1;
      end
   end
```

```
end


distance = sqrt( distance / ((size_data(1))-1));
```

A.5 Sorting Matrices

```
function [sort_dist_matrix, sort_resi_matrix] = ...
 sort_matrices_bydistance(sort_dist_matrix, sort_resi_matrix, size_train)


for i = 1 : size_train


    temp(:,1) = sort_dist_matrix(:,i);
    temp(:,2) = sort_resi_matrix(:,i);

    temp = sortrows(temp,1);


    sort_dist_matrix(:,i) = temp(:,1);
    sort_resi_matrix(:,i) = temp(:,2);


end
```

A.6 Calculating Mantel's Correlation

```
function mantels_corr = calc_mantels_corr(dist_matrix, resi_matrix,...
    size_train)


correlation_value = corrcoef(dist_matrix, resi_matrix);
```

```
mantels_corr = correlation_value(1,2);
```

A.7 Calculating Probability from Case Profile

```
function [probability] = calc_probability(i, quad, distances, cb_profile)

if quad == 2

   if(distances(i,3) <= 0.25)
      if(cb_profile(distances(i,1),1)+cb_profile(distances(i,1),5)...
            +cb_profile(distances(i,1),9)+cb_profile(distances(i,1),13 ...
          )) == 0
         probability = 1;
      else
         probability = ((cb_profile(distances(i,1),1)+cb_profile...
             (distances(i,1),5)) / (cb_profile(distances(i,1),1)+...
             cb_profile(distances(i,1),5)+cb_profile(distances(i,1),9)+...
             cb_profile(distances(i,1),13)));
      end

   elseif(distances(i,3) <= 0.5)
      if(cb_profile(distances(i,1),2)+cb_profile(distances(i,1),6)+...
            cb_profile(distances(i,1),10)+cb_profile(distances(i,1),14)) == 0
         probability = 1;
      else
         probability = ((cb_profile(distances(i,1),2)+cb_profile...
             (distances(i,1),6)) / (cb_profile(distances(i,1),2)+...
             cb_profile(distances(i,1),6)+cb_profile(distances(i,1),10)+...
             cb_profile(distances(i,1),14)));
      end

   elseif(distances(i,3) <= 0.75)
```

```
if(cb_profile(distances(i,1),3)+cb_profile(distances(i,1),7)+...
        cb_profile(distances(i,1),11)+cb_profile(distances(i,1),15 ...
    )) == 0
   probability = 1;
else
    probability = ((cb_profile(distances(i,1),3)+cb_profile...
        (distances(i,1),7)) / (cb_profile(distances(i,1),3)+...
        cb_profile(distances(i,1),7)+cb_profile(distances(i,1),11)+...
        cb_profile(distances(i,1),15)));
  end

elseif(distances(i,3) <= 1)
  if(cb_profile(distances(i,1),4)+cb_profile(distances(i,1),8)+...
        cb_profile(distances(i,1),12)+cb_profile(distances(i,1),16 ...
         )) == 0
    probability = 1;
  else
      probability = ((cb_profile(distances(i,1),4)+cb_profile...
          (distances(i,1),8)) / (cb_profile(distances(i,1),4)+...
          cb_profile(distances(i,1),8)+cb_profile(distances(i,1),12)+...
          cb_profile(distances(i,1),16)));
    end
  end

elseif quad == 1

  if(distances(i,3) <= 0.25)
    if(cb_profile(distances(i,1),1)+cb_profile(distances(i,1),5)+...
          cb_profile(distances(i,1),9)+cb_profile(distances...
          (i,1),13)) == 0
    probability = 1;
  else
    probability = ((cb_profile(distances(i,1),1)) / (cb_profile...
        (distances(i,1),1)+cb_profile(distances(i,1),5)+cb_profile...
        (distances(i,1),9)+cb_profile(distances(i,1),13)));
```

```
        end

    elseif(distances(i,3) <= 0.5)
        if(cb_profile(distances(i,1),2)+cb_profile(distances(i,1),6)+...
                cb_profile(distances(i,1),10)+cb_profile(distances(i,1)...
                ,14)) == 0
            probability = 1;
        else
            probability = ((cb_profile(distances(i,1),2)) / (cb_profile...
                (distances(i,1),2)+cb_profile(distances(i,1),6)+cb_profile...
                (distances(i,1),10)+cb_profile(distances(i,1),14)));
        end

    elseif(distances(i,3) <= 0.75)
        if(cb_profile(distances(i,1),3)+cb_profile(distances(i,1),7)+...
                cb_profile(distances(i,1),11)+cb_profile(distances...
                (i,1),15)) == 0
            probability = 1;
        else
            probability = ((cb_profile(distances(i,1),3)) / (cb_profile...
                (distances(i,1),3)+cb_profile(distances(i,1),7)+cb_profile...
                (distances(i,1),11)+cb_profile(distances(i,1),15)));
        end

    elseif(distances(i,3) <= 1)
        if(cb_profile(distances(i,1),4)+cb_profile(distances(i,1),8)+...
                cb_profile(distances(i,1),12)+cb_profile(distances...
                (i,1),16)) == 0
            probability = 1;
        else
            probability = ((cb_profile(distances(i,1),4)) / (cb_profile...
            (distances(i,1),4)+cb_profile(distances(i,1),8)+cb_profile...
            (distances(i,1),12)+cb_profile(distances(i,1),16)));
        end
    end
```

```
end
```

A.8 Predicting Solutions

```
function [cases_k, solutions_k, prediction_k, error_k] = predict_k ...
    (k, train, target, distances)

colnames = fieldnames(train);
prediction_k = 0;
error_k = 0;
cases_k = zeros(k);
solutions_k = zeros(k);


for i = 1 : k
   prediction_k = prediction_k + train(distances(i,1)).(colnames{1});
   solutions_k(i) = train(distances(i,1)).(colnames{1});
   cases_k(i) = distances(i,1);
end

prediction_k = round(prediction_k / k);
error_k = abs( target.(colnames{1})-prediction_k);


function [cases_p, solutions_p, prediction_p, error_p] = ...
    predict_p(k, p, quad, train, target, distances, cb_profile)

colnames = fieldnames(train);
```

A Script Listing - Main.m

```
prediction_p = 0;
error_p = 0;


cases_p = zeros(k);
solutions_p = zeros(k);

k_counter = 0;
i = 1;

while k_counter < k

   probability = calc_probability(i, quad, distances, cb_profile);

   if probability >= p

      prediction_p = prediction_p + train(distances(i,1)).(colnames{1});
      solutions_p(k_counter + 1) = train(distances(i,1)).(colnames{1});
      cases_p(k_counter + 1) = distances(i,1);

      i = i + 1;
      k_counter = k_counter + 1;

   else
      i = i + 1;
   end
end

prediction_p = round(prediction_p / k);
error_p = abs( target.(colnames{1})-prediction_p);
```

VI

Bibliography

PAGE

MISSING

IN

ORIGINAL

Bibliography

[AB94] D. W. Aha and R. L. Bankert. Feature Selection for Case-Based Classification of Cloud Types: An Empirical Comparison. In D. W. Aha, editor, *Case-Based Reasoning: Papers from the 1994 Workshop*, Menlo Park, CA, 1994. AAAI.

[Aha91] D. W. Aha. Case–Based Learning Algorithms. In R. Bareiss, editor, *Procs. of the 1991 DARPA Case–Based Reasoning Workshop*, pages 147–158, San Francisco, 1991. Morgan Kaufmann.

[Aha97] D. W. Aha. *Artificial Intelligence Review - Special Issue on Lazy Learning*, volume 11. Springer, 1997.

[Aha98] D. W. Aha. The Omnipresence of Case-Based Reasoning in Science and Application. *Knowledge-Based Systems*, 11(5-6):261–273, 1998.

[AKA91] D. W. Aha, D. Kibler, and M. K. Albert. Instance–Based Learning Algorithms. *Machine Learning*, 6:37–66, 1991.

[Alt89] K. D. Althoff. Knowledge Acqusition in the Domain of CNC Machine Centers: The MOLTKE Approach. In J. Boose, B. Gaines, and J.-G. Ganascia, editors, *In Procs. of the Third European Workshop on Knowledge-Based Systems, EKAW–89*, pages 180–195, Paris, France, 1989.

[Alt01] K. D. Althoff. Case–Based Reasoning. In *Handbook of Software Engineering and Knowledge Engineering*, volume 1. World Scientific Pub. Co., 2001.

[AP94] A. Aamodt and E. Plaza. Case–Based Reasoning: Foundation Issues, Methodological Variations, and System Approaches. *AI Communication*, 7(1):39–59, 1994.

[Arm02] P. Armour. Ten unmyths of project estimation. *Communications of the ACM*, 42(11):15–18, November 2002.

[AS00] L. Angelis and I. Stamelos. A Simulation Tool for Efficient Analogy Based Cost Estimation. *Empirical Software Engineering*, 5:35–68, 2000.

[BCHW95] B. W. Boehm, B. Clark, E. Horowitz, and C. Westland. Cost Models for Future Life Cycle Processes: COCOMO 2.0. *Annals of Software Engineering*, 1:57–94, 1995.

[BES+99] L. C. Briand, K. E. Emam, D. Surmann, I. Wieczorek, and K. Maxwell. An Assessment and Comparison of Common Software Cost Estimation Modeling Techniques. In *Procs. of the 21st International Conference for Software Engineering, ICSE–1999*, pages 313–322, Los Angeles, USA, 1999. ACM.

[BM95] R. Bisio and F. Malabocchia. Cost Estimation of Software Projects through Case Base Reasoning. In M. M. Veloso and A. Aamodt, editors, *Case-Based Reasoning Research and Development: 1st International Conference, ICCBR–95*, volume 1010 of *LNCS*, pages 11–22, Sesimbra, Portugal, 1995. Springer.

[BM02] H. Brighton and C. Mellish. Advances in Instance Selection for Instance-Based Learning Algorithms. *Data Mining and Knowledge Discovery*, 6(2):153–172, April 2002.

[Boe81a] B. Boehm. *Software Engineering Economics*. Prentice Hall, 1981.

[Boe81b] B. W. Boehm. *Software Engineering Economics*. Prentice Hall, Englewood Cliffs, NJ, USA, 1981.

[BTGCDA04] J. J. Bello-Tomás, P. A. González-Calero, and B. Díaz-Agudo. JColibri: an Object-Oriented Framework for Building CBR Systems. In P. Funk and P.A. González-Calero, editors, *Advances in Case-Based Reasoning: 7th European Conference, ECCBR–04*, volume 3155 of *LNAI*. Springer, 2004.

[BW01] L. C. Briand and I. Wieczorek. Resource Estimation in Software Engineering. Technical Report ISERN 00-05, Carleton University and Fraunhofer Institute for Experimental Software Engineering, 2001.

[CBS97] P. Cunningham, A. Bonzano, and B. Smyth. Using Introspective Learning to Improve Retrieval in Car: A Case Study in Air Traffic Control. In

D. Leake and E. Plaza, editors, *Case-Based Reasoning Research and Development: Second International Conference, ICCBR-97*, volume 1266 of *LNCS*, pages 291–302, Providence, Rhode Island, USA, 1997. Springer.

[Che00] W. Cheetham. Case–Based Reasoning with Confidence. In E. Blanzieri and L. Portinale, editors, *Advances in Case–Based Reasoning: 5th European Workshop, EWCBR–00*, volume 1898 of *LNCS*, pages 15–25, Trento, Italy, 2000. Springer-Verlag.

[CL05] P.-C. Chang and C.-Y. Lai. A Hybrid System Combining Self-Organizing Maps with Case-Based Reasoning in Wholesaler's New-Release Book Forecasting. *Expert Systems with Applications*, 29:183–192, 2005.

[CP04] W. Cheetham and J. Price. Measures of Solution Accuracy in Case–Based Reasoning Systems. In P. Funk and P. A. González-Calero, editors, *Advances in Case–Based Reasoning: 7th European Conference, ECCBR–04*, volume 3155 of *LNCS*, pages 106–118, Madrid, Spain, 2004. Springer-Verlag.

[Cra03] S. Craw. Instrospective Learning to Build Case-Based Reasoning (CBR) Knowledge Containers. In *IAPR International Workshop on Machine Learning and Data Mining in Pattern Recognition, MLDM–2003*, pages 1–6. Springer, 2003.

[Cun98] P. Cunningham. CBR: Strenghts and Weaknesses. In A. P. d. Pobil, J. Mira, and M. Ali, editors, *Procs. of the 11th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, volume 2 of *LNAI 1416*, pages 517–523. Springer Verlag, 1998.

[DC00] S. J. Delany and P. Cunningham. The Application of Case-Based Reasoning to Early Software Project Cost Estimation and Risk Assessment. Technical Report TCD-CS-2000-10, Trinity College, Dublin, Ireland, 2000.

[DC04] S. J. Delany and P. Cunningham. An Analysis of Case-Base Editing in a Spam Filtering System. In P. Funk and P. A. González-Calero, editors, *Advances in Case-Based Reasoning: 7th European Conference, ECCBR–2004*, volume 3155 of *LNCS*, pages 128–141, Madrid, Spain, 2004. Springer.

[DCW98] S. J. Delany, P. Cunningham, and W. Wilke. The Limits of CBR in Software Project Estimation. In *Procs. of the 6th German Workshop on Case–Based Reasoning*, Berlin, Germany, 1998.

[Del98] S. J. Delany. The Design of a Case Representation for Early Software Development Cost Estimation. Master's thesis, Stafford University, UK, 1998.

[Des89] J. M. Desharnais. Analyse Statistique de la Productivite des Projets de Developpement en Informatique a Partir de la Technique des Points de Fonction. Master's thesis, University of Montreal, Quebec, Canada, 1989.

[DLM99] J. Daengdej, D. Lukose, and R. Murison. Using Statistical Models and Case-Based Reasoning in Claims Prediction: Experience from a Real-World Problem. *Knowledge-Based Systems*, 12(5):239–245, October 1999.

[DLT+97] J. Daengdej, D. Lukose, E. Tsui, P. Beinat, and L. Prophet. Combining Case–Based Reasoning and Statistical Method for Proposing Solution in RICAD. *Knowledge Based Systems*, 10:153–159, 1997.

[EA01] A. M. Essam and S. M. Ahmed. Applying Neural Networks in Case–Based Reasoning Adaptation for Cost Assessment of Steel Buildings. In *Procs. of the Euro-International Symposium on Computational Intelligence*, Slovakia, 2001.

[EN03] R. Elmasri and S. B. Navathe. *Fundamentals of Database Systems*. Pearsons Education, fourth edition, 2003.

[Exp] Experience Pro Internet Site. www.sttf.fi.

[FPPA91] D. Freedman, R. Pisani, R. Purves, and A. Adhikari. *Statistics*. W. W. Norton and Company, New York, second edition, 1991.

[FR93] A. G. Francis and A. Ram. The Utility Problem in Case–Based Reasoning. In *Case–Based Reasoning: Papers from the AAAI Workshop Technical Report WS-93-01*, Wasington D.C., USA, 1993. AAAI Press.

[FS02] G. R. Finnie and Z. Sun. Similarity and metrics in case-based reasoning. *International Journal of Intelligent Systems*, 17(3):273–287, 2002.

[FW97] G. R. Finnie and G. E. Wittig. A Comparison of Software Effort Estimation Techniques: Using FPs with Neural Networks, Case Based Reasoning and Regression Models. *Journal of Systems and Software*, page 9, 1997.

[FW03] G. R. Finnie and G. E. Wittig. R^5 model for Case-Based Reasoning. *Knowledge-Based Systems*, 16(1):59–65, January 2003.

[FWD97] G. R. Finnie, G. E. Wittig, and J. M. Desharnais. Estimating Software Development Effort with Case-Based Reasoning. In D. Leake and E. Plaza, editors, *Case-Based Reasoning Research and Development, 2nd International Conference, ICCBR-97*, volume 1266 of *LNCS*, pages 13–22, Providence, Rhode Island, USA, 1997. Springer.

[Gat72] G. W. Gates. The Reduced Nearest Neighbour Rule. *IEEE Transations on Information Theory*, 18(3):4, 431-433 1972.

[Har68] P. E. Hart. The Condensed Nearest Neighbour Rule. *IEEE Transactions on Information Theory*, 14(3):515–516, 1968.

[Hee92] F. J. Heemstra. Software Cost Estimation. *Information and Software Technology*, 34(10):627–639, 1992.

[HI96] J. Hongkyu and H. Ingoo. Integration of Case–Based Forecasting, Neural Network and Discriminant Analysis for Bankruptcy Prediction. *Expert Systems with Applications*, 11(4):415–422, 1996.

[Hug96] R. T. Hughes. An evaluation of machine learning techniques for software cost estimation. Technical report, University of Brighton, UK, 1996.

[IA01] A. Idri and A. Abran. A Fuzzy Logic Based Set of Measures for Software Project Similarity: Validation and Possible Improvements. In *Procs. of the 7th IEEE International Software Metrics Symposium, METRICS–2001*, pages 85–97, London, UK, 2001. IEEE Computer Society.

[IAK02] A. Idri, A. Abran, and T. M. Khoshgoftaar. Estimating software project effort by analogy based on linguistic values. In *Procs. of the 8th IEEE International Software Metrics Symposium, METRICS–2002*, pages 21–, Ottawa, Canada, 2002. IEEE Computer Society.

[IRB00] I. Iglezakis and T. Roth-Berghofer. A Survey Regarding the Central Role of the Case Base for Maintenance in Case–Based Reasoning. In *Procs. of the ECAI Workshop on Flexible Strategies for Maintaining Knowledge Containers*, pages 22–28, Berlin, Germany, 2000.

[IRRB04] I. Iglezakis, T. Reinartz, and T. Roth-Berghofer. Maintenance Memories: Beyond Concepts and Techniques for Case Base Maintenance. In P. Funk and P. A. González-Calero, editors, *Advances in Case-Based Reasoning: 7th European Conference, ECCBR-2004*, volume 3155 of *LNCS*, pages 227–241, Madrid, Spain, 2004. Springer.

[J.99] Crichton N. J. Information Point: Spearman's Rank Correlation. *Journal of Clinical Nursing*, 8:763, 1999.

[JIS03] M. Jørgensen, U. Indahl, and D. Sjøberg. Software Effort Estimation by Analogy and Regression toward the Mean. *Journal of Systems and Software*, 68(3):253–262, 2003.

[Jør04] M. Jørgensen. A Review of Studies on Expert Estimation of Software Development Effort. *Journal of Systems and Software*, 70(1–2):37–60, 2004.

[JW99] R. Jeffery and F. Walkerden. Analogy, Regression and Other Methods for Estimating Effort and Software Quality Attributes. In R. Kusters, A. Cowderoy, and E. v. Veenendaal, editors, *Procs. of the 10th Conference on European Software Control and Metrics*, Herstmonceux Castle, United Kingdom, 1999. Shaker Publishing.

[KA04] Z. Kira and R. C. Arkin. Forgetting Bad Behavior: Memory Management for Case-Based Navigation. In *Procs. of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS-04*, Sendai, Japan, 2004. IEEE.

[Kan99] G. K. Kanji. *100 Statistical Tests*. SAGE Publications, second edition, 1999.

[KB93] M. Kriegsman and R. Barletta. Building a Case-Based Help Desk Application. *IEEE Expert: Intelligent Systems and Their Applications*, 8(6):18–26, 1993.

[KCCS00] G. Kadoda, M. Cartwright, L. Chen, and M. J. Shepperd. Experiences Using Case-Based Reasoning to Predict Software Project Effort. In *Procs. of the 4th International Conference on Empirical Assessment & Evaluation in Software Engineering*, Keele University, Staffordshire, UK, 2000.

[KCS00] G. Kadoda, M. Cartwright, and M. J. Shepperd. On Configuring a Case-Based Reasoning Software Project Prediction System. In *UK CBR Workshop*, Cambridge, UK, 2000.

[Kem87] C. F. Kemerer. An Empirical Validation of Software Cost Models. *Communications of the ACM*, 30(5):416–429, 1987.

[KH01] K. S. Kim and I. Han. The Cluster-Indexing Method for Case–Based Reasoning Using Self-Organising Maps and Learning Vector Quantization for Bond Rating Cases. *Expert Systems with Applications*, 21:147–156, 2001.

[Kit97] B. A. Kitchenham. Counterpoint: The Problem with Function Points. *IEEE Software*, 14(2):29–31, 1997.

[KKJ05] J. Keung, B. Kitchenham, and R. Jeffery. 'Analogy-X' - An Extension to Cost Estimation Using Analogy. Technical Report Unpublished Report, NICTA, Sydney, Australia, 2005.

[Kol93] J. Kolodner. *Case-Based Reasoning*. Morgan Kaufmann Publishers, Inc., San Francisco, CA, 1993.

[KPMS01] B. Kitchenham, L. M. Pickard, S. G. MacDonell, and M. Shepperd. What Accuracy Statistics Really Measure. *IEE Proceedings - Software*, 148(3):81–85, 2001.

[KPS03] Mendes-E. Kirsopp, C., R. Premraj, and M. Shepperd. An empirical analysis of linear adaptation techniques for case-based prediction. In K. D. Ashley and D. G. Bridge, editors, *Case-Based Reasoning Research and Development: 5th International Conference, ICCBR–2003*, volume 2689, pages 231–245, Trondheim, Norway, 2003. Springer.

[KS02a] C. Kirsopp and M. Shepperd. Case and Feature Subset Selection in Case-Based Software Project Effort Prediction. In *Procs. of the 22nd SGAI International Conference on Knowledge Based Systems and Applied Artificial Intelligence, ES–2002*, pages 61–74, Cambridge, 2002. Springer.

[KS02b] C. Kirsopp and M. Shepperd. Making Inferences with Small Numbers of Training Sets. *IEE Proceedings - Software*, 149(5):123–130, 2002.

[KSH02] C. Kirsopp, M. Shepperd, and J. Hart. Search Heuristics, Case-based Reasoning And Software Project Effort Prediction. In *Procs. of the Genetic*

and Evolutionary Computation Conference, GECCO-2002, pages 1367–1374, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc.

[Lea96] D. B. Leake. *CBR in Context: The Present and Future*, chapter 1. AAAI Press/MIT Press, Menlo Park, 1996.

[LF02] H. Leung and Z. Fan. Software Cost Estimation. In S. K. Chang, editor, *Handbook of Software Engineering and Knowledge Engineering*, volume 2, page 808. World Scientific, 2002.

[LS01] D. B. Leake and R. Sooriamurthi. When Two Case Bases Are Better Than One: Exploiting Multiple Case Bases. In *Case-Based Reasoning Research and Development: 4th International Conference, ICCBR-2001*, pages 321–335, Vancouver, Canada, 2001. Springer Verlag.

[LS02a] D. B. Leake and R. Sooriamurthi. Automatically Selecting Strategies for Multi-Case-Base Reasoning. In *Advances in Case-Based Reasoning: 6th European Workshop, EWCBR-2002*, pages 204–218, Aberdeen, Scotland, 2002. Springer Verlag.

[LS02b] D. B. Leake and R. Sooriamurthi. Managing Multiple Case Bases: Dimensions and Issues. In *Procs. of 15th International Florida Artificial Intelligence Research Society Conference, FLAIRS-2002*, pages 106–110, Pensacola Beach, Florida, 2002. AAAI Press.

[LW98] D. B. Leake and D. C. Wilson. Categorizing Case-Base Maintenance: Dimensions and Directions. In B. Smyth and P. Cunningham, editors, *Advances in Case-Based Reasoning: 4th European Workshop, EWCBR-98*, volume 1488 of *LNCS*, pages 196–207, Dublin, Ireland, 1998. Springer-Verlag.

[LW99] D. Leake and D. C. Wilson. When Experience is Wrong: Examining CBR for Changing Tasks and Environments. In K.-D. Althoff, R. Bergmann, and K. Branting, editors, *Case-Based Reasoning Research and Development: 3rd International Conference, ICCBR-99*, volume 1650 of *Lecture Notes in Computer Science*, pages 218–232, Seeon Monastery, Germany, 1999. Springer.

[Man01] B. F. J. Manly. *Randomisation, Bootstrap and Monte Carlo Methods in Biology*. Texts in Statistical Science Series. Chapham & Hall/CRC, second edition, 2001.

[MAT] MathWorks Internet Site. www.mathworks.com.

[MBM94] J. E. Matson, B. E. Barrett, and J. M. Mellichamp. Software Development Cost Estimation Using Function Points. *IEEE Transactions on Software Engineering*, 20:275–287, 1994.

[MCM02] E. Mendes, S. Counsell, and N. Mosley. A Comparison of Case-Based Reasoning Approaches to Web Hypermedia Project Cost Estimation. In *Procs. of the 11th International Conference on World Wide Web*, pages 272–280, Honolulu, Hawaii, USA, 2002. ACM Press.

[MCM03] E. Mendes, S. Counsell, and N. Mosley. Investigating the Use of Case-Based Reasoning Adaptation Rules for Web Project Cost Estimation. In *Procs. of the 12th International Conference on World Wide Web*, Budapest, Hungary, 2003. ACM Press.

[MF00] K. Maxwell and P. Forselius. Benchmarking Software Development Productivity. *IEEE Software*, January-February:80–88, 2000.

[MG96] S. G. MacDonnel and A. R. Gray. Alternatives to regression models for estimation software prokects. In *IFPUG Fall Conference*, Dallas, USA, 1996.

[MK92] T. Mukhopadhyay and S. Kekre. Software Effort Models for Early Estimation of Process Control Applications. *IEEE Transactions of Software Engineering*, 18(10):915–924, October 1992.

[MK04] E. Mendes and B. A. Kitchenham. Further Comparison of Cross-Company and Within-Company Effort Estimation Models for Web Applications. In *Procs. of the 10th IEEE International Software Metrics Symposium, METRICS-2004*, pages 348–357. IEEE Computer Society, 2004.

[MKL+00] C. Mair, G. Kadoda, M. Lefley, K. Phalp, C. Schofield, M. Shepperd, and S. Webster. An Investigation of Machine Learnning Based Prediction Systems. *Journal of Systems and Software*, 53(1):23–29, 2000.

[MLdeR02] M. Montaner, B. López, and J. L. d. e. Rosa. Improving Case Representation and Case Base Maintenance in Recommender Agents. In S. Craw and A. Preece, editors, *Advances in Case-Based Reasoning: 6th European Workshop, EWCBR-2002*, pages 234–248, Aberdeen, Scotland, 2002. Springer-Verlag.

[MS99] I. Myrtveit and E. Stensrud. A Controlled Experiment to Assess the Benefits of Estimaing with Analogy and Regression Models. *IEEE Transactions on Software Engineering*, 25(4):510–525, 1999.

[MS00] E. McKenna and B. Smyth. Competence-Guided Case-Base Editing Techniques. In E. Blanzieri and L. Portinale, editors, *Advances in Case–Based Reasoning: 5th European Workshop, EWCBR–00*, volume 1898 of *LNCS*, pages 186–197, Trento, Italy, 2000. Springer.

[MS01a] E. McKenna and B. Smyth. An Interactive Visualisation Tool for Case–Based Reasoners. *Applied Intelligence*, 14(1):95–114, 2001.

[MS01b] M. Mullins and B. Smyth. Visualisation Methods in Case-Based Reasoning. In D. W. Aha and I. Watson, editors, *Case-Based Reasoning Research and Development: 4th International Conference, ICCBR–2001, Workshop On Case–Based Reasoning Authoring Support Tools*, volume 2080 of *LNCS*, Vancouver, Canada, 2001. Springer–Verlag.

[MS05] C. Mair and M. Shepperd. The Consistency of Empirical Comparisons of Regression and Analogy-based Software Project Cost Prediction. In *Procs. of the 4th IEEE International Symposium on Empirical Software Engineering*, Noosa Heads, Australia, 2005.

[MsJ03] K. J. Moløkken-Østvold and M. Jørgensen. A Review of Surveys on Software Effort Estimation. In *IEEE International Symposium on Empirical Software Engineering, ISESE–2003*, pages 223–230, Rome, Italy, 2003. IEEE Computer Society.

[MSJ05] C. Mair, M. Shepperd, and M. Jørgensen. An Analysis of Data Sets Used to Train and Validate Cost Prediction Systems. In *Procs. of the 2005 Workshop on Predictor Models in Software Engineering, PROMISE–2005*, pages 1–6, New York, NY, USA, 2005. ACM Press.

[MWT+03] E. Mendes, I. Watson, C. Triggs, N. Mosley, and S. Counsell. A Comparative Study of Cost Estimation Models for Web Hypermedia Applications. *Empirical Software Engineering*, 8(2):163–196, 2003.

[NV97] F. Niessink and H. v. Vliet. Predicting Maintenance Effort with Function Points. In *13th International Conference on Software Maintenance , ICSM–97*, page 32, Bari, Italy, 1997.

[PS03] U. Passing and M. Shepperd. An Experiment on Software Project Size and Effort Estimation. In *IEEE International Symposium on Empirical Software Engineering, ISESE-2003*, pages 120–131, Rome, Italy, 2003. IEEE Computer Society.

[PS04] S. K. Pal and S. C. K. Shiu. *Foundations of Soft Case-Based Reasoning*. Wiley Series on Intelligent Systems. John Wiley & Sons, New Jersey, 2004.

[PSC03] R. Premraj, M. Shepperd, and M Cartwright. Meta-data to Guide Retrieval in CBR for Software Cost Prediction. In B. Lees, editor, *Procs. of the 8th UK Workshop on Case-Based Reasoning*, pages 26–37, Cambridge, UK, 2003.

[PSKF05] R. Premraj, M. Shepperd, B. Kitchenham, and P. Forselius. An Empirical Analysis of Software Productivity over Time. In *11th IEEE International Software Metrics Symposium*, Como, Italy, 2005. IEEE Computer Society.

[PT00] L. Portinale and P. Torasso. Automatic Case Base Management in a Multi-Modal Reasoning System. In L. Portinale and E. Blanzieri, editors, *Advances in Case-Based Reasoning, 5th European Workshop, EWCBR-00*, volume 1898 of *LNCS*, Trento, Italy, 2000. Springer.

[PTT99] L. Portinale, P. Torasso, and P. Tavano. Speed-up, Quality and Competence in Multi-Modal Case-Based Reasoning. In K.-D. Althoff, R. Bergmann, and L. K. Branting, editors, *Case-Based Reasoning and Development, 3rd International Conference, ICCBR-99*, volume 1650 of *LNCS*, pages 303–317, Seeon Monastery, Germany, 1999. Springer-Verlag.

[Put78] L. Putnam. A general empirical solution to the macro software sizing and estimation problem. *IEEE Transactions of Software Engineering*, 4(4):345–361, 1978.

[PVM96] M. J. Prietula, S. Vicinanza, and T. Mukhopadhyay. Software Effort Estimation with a Case-Based Reasoner. *Journal of Expert and Theoritical Artificial Intelligence*, 8:341–363, 1996.

[RB03] T. Roth-Berghofer. *Knowledge Maintenance of Case-Based Reasoning Systems. The SIAM methodology*. PhD thesis, Akademische Verlagsgesellschaft Aka GmbH, DISKI 262, Berlin, Germany, 2003.

[RBI01] T. Roth-Berghofer and I. Iglezakis. Six Steps in Case-Based Reasoning: Towards a Maintenance Methodology for Case-Based Reasoning Systems. In H.-P. Schnurr, R. Staab, S. Studer, G. Stumme, and Y. Sure, editors, *Procs. of the 9th German Workshop on Case-Based Reasoning*, pages 198–208, Baden, Germany, 2001. Shaker-Verlag.

[RBR01] T. Roth-Berghofer and T. Reinartz. MAMA: A Maintenance Manual for Case-Based Reasoning Systems. In D. W. Aha and I. Watson, editors, *Case-Based Reasoning Research and Development: 4th International Conference, ICCBR-01*, volume 2080 of *LNCS*, pages 452–466, Vancouver, Canada, 2001. Springer.

[Ric98] M. Richter. Introduction. In M. Lenz, B. Bartsch-Sproll, H-D. Burkhard, and S. Wess, editors, *CBR Technology: From Foundations to Applications*, volume 1400 of *LNAI*, pages 1–15. Springer-Verlag, Berlin, Germany, 1998.

[RIRB00] T. Reinartz, I. Iglezakis, and T. Roth-Berghofer. On Quality Measures for Case Base Maintenance. In E. Blanzieri and L. Portinale, editors, *Advances in Case-Based Reasoning: 5th European Workshop, EWCBR-00*, volume 1898 of *LNCS*, pages 247–259, Trento, Italy, 2000. Springer.

[RIRB01] T. Reinartz, I. Iglezakis, and T. Roth-Berghofer. Review and Restore for Case Base Maintenance. *Computational Intelligence*, 17(2), 2001.

[RWLI75] G. L. Ritter, H. B. Woodruff, S. R. Lowry, and T. L. Isenhour. An Algorithm for a Selective Nearest Neighbour Decision Rule. *IEEE Transactions on Information Theory*, 21(6):665–669, 1975.

[RY96] K. Racine and Q. Yang. On the Consistency Management of Large Case Bases: The Case for Validation. In *Procs. of the AAAI-96 Workshop on Knowledge Base Validation, American Association for Artificial Intelligence, AAAI-1996*, 1996.

[RY97] K. Racine and Q. Yang. Maintaining Unstructured Case Base. In D. Leake and E. Plaza, editors, *Case-Based Reasoning Research and Development: Second International Conference, ICCBR-97*, volume 1266 of *LNCS*, pages 553–564, Providence, Rhode Island, USA, 1997. Springer.

[RY01] K. Racine and Q. Yang. Redundancy Detection in Semistructured Case Bases. *IEEE Transactions on Knowledge and Data Engineering*, 13(3):513–518, 2001.

[SC96] B. Smyth and P. Cunningham. The Utility Problem Analysed - a Case Based Reasoning Perspective. In I. F. C. Smith and B. Faltings, editors, *Advances in Case–Based Reasoning: 3rd European Workshop, EWCBR–96*, volume 1168 of *LNCS*, pages 392–399, Lausanne, Switzerland, 1996. Springer.

[Sca94] W. Scacchi. Understanding Software Productivity. In D. Hurley, editor, *Advances in Software Engineering and Knowledge Engineering*, volume 4, pages 37–70, 1994.

[SCA05] F. Sørmo, J. Cassens, and A. Aamodt. Explanation in Case-Based Reasoning – Perspectives and Goals. *Artificial Intelligence Review*, 24(2):109–143, October 2005.

[Sch82] R. Schank. *Dynamic Memory: A Theory of Learning in Computers and People*. Cambridge University Press, Cambridge, UK, 1982.

[Sch98] C. Schofield. *An Empirical Investigation into Software Effort Estimation by Analogy*. PhD thesis, Bournemouth University, United Kingdom, 1998.

[SF95] K. Srinivasan and D. Fisher. Machine Learning Aproaches to Estimating Software Development Effort. *IEEE Transactions on Software Engineering*, 21(2):126–136, 1995.

[SG04] M. Salamó and E. Golobardes. Dynamic Case Base Maintenance for a Case–Based Reasoning System. In C. Lemaître, C. A. Reyes, and J. u. s. A. González, editors, *Advances in Artificial Intelligence - IBERAMIA–04, 9th Ibero-American Conference on AI*, volume 3315, pages 93–103, Puebla, México, 2004. Springer-Verlag.

[SK94] B. Smyth and M. T. Keane. Retrieving Adaptable Cases: The Role of Adaptation Knowledge in Case Retrieval. In *EWCBR '93: Selected papers from the First European Workshop on Topics in Case-Based Reasoning*, pages 209–220, London, UK, 1994. Springer-Verlag.

[SK95] B. Smyth and M. T. Keane. Remembering to Forget: A Competence-Preserving Case Deletion Policy for Case–Based Reasoning Systems. In

Procs. of the 14th International Joint Conference on Artificial Intelligence, IJCAI–95, pages 377–383, Montréal, Québec, Canada, 1995.

[SK01] M. Shepperd and G. Kadoda. Comparing Software Prediction Techniques Using Simulation. *IEEE Transactions on Software Engineering*, 27(11):1–9, November 2001.

[Ska94] D. B. Skalak. Prototype and Feature Selection by Sampling and Random Mutation Hill Climbing Algorithms. In *Procs. of the 11th International Machine Learning Conference, ICML–1994*, pages 293–301, New Brunswick, NJ, 1994. Morgan Kauffmann.

[SM98] B. Smyth and E. McKenna. Modelling the Competence of Case-Bases. In B. Smyth and P. Cunningham, editors, *Advances in Case-Based Reasoning: 4th European Workshop, EWCBR–98*, volume 1488 of *LNCS*, pages 208–220, Dublin, Ireland, 1998. Springer-Verlag.

[SM99] B. Smyth and E. McKenna. Building Compact Competent Case–Bases. In K-D. Althoff, R. Bergmann, and K. Branting, editors, *Case-Based Reasoning Research and Development: 3rd International Conference, ICCBR–99*, volume 1650 of *LNAI*, pages 329–342, Seeon Monastery, Germany, 1999. Springer-Verlag.

[SM01] B. Smyth and E. McKenna. Competence Guided Incremental Footprint-Based Retrieval. *Knowledge Based Systems*, 14:155–161, 2001.

[SS97] M. Shepperd and C. Schofield. Estimating Software Project Effort Using Analogies. *IEEE Transactions on Software Engineering*, 23:736–743, 1997.

[SSK96] M. Shepperd, C. Schofield, and B. Kitchenham. Effort Estimation Using Analogy. In *Procs. of the 18th International Conference on Software Engineering, ICSE–1996*, pages 170–178, Berlin, Germany, 1996. IEEE Computer Society Press.

[Tad05] N. Tadayon. Neural Network Approach for Software Cost Estimation. In *Procs. of the International Conference on Information Technology: Coding and Computing (ITCC'05) - Volume II*, pages 815–818, 2005.

[TC05] B. Twala and M. Cartwright. Ensemble Imputation Methods for Missing Software Engineering Data. In *Procs. of the 11th International Symposium*

on *Software Metrics*, *IEEE METRICS-2005*, page 30, Como, Italy, 2005. IEEE.

[Tom76] I. Tomek. An Experiment with the Nearest Neighbour Rule. *IEEE Transactions on Systems, Man, and Cybernetics*, 6(6):448–452, 1976.

[Ven93] A. R. Venkatachalam. Software Cost Estimation Using Artificial Neural Networks. In *Procs. of the International Joint Conference on Neural Networks*, Nagoya, 1993. IEEE.

[Ves99] P. Vesterinen. Issues in Calibrating Effort Estimation Models. *Software Engineering Notes*, 24(3):63–65, May 1999.

[VMP91] S. Vicinanza, T. Mukhopadhyay, and M. Prietula. Software Effort Estimation: An Exploratory Study of Expert Performance. *Information Systems Research*, 2(4):243–262, 1991.

[VO96] A. Voss and R. Oxman. A Study of Case Adaptation Systems. In J. S. Gero and F. Sudweeks, editors, *Artificial Intelligence in Design*, pages 173–189, Stanford, USA, 1996. Kluwer Academic Publishers.

[VPM90] S. Vicinanza, M. J. Prietula, and T. Mukhopadhyay. Case-Based Reasoning in Software Effort Estimation. In *11th International Conference on Information Systems*, 1990.

[Wat98] I. Watson. CBR is a Methodology Not a Technology. In R. Miles, M. Moulton, and M. Bramer, editors, *Research & Development in Expert Systems*, volume 15, pages 213–223, London, UK, 1998. Springer-Verlag.

[Wil72] D. L. Wilson. Asymptotic Properties of Nearest Neighbor Rules Using Edited Data. *IEEE Transactions on Systems, Man, and Cybernetics*, 2(3):408–421, 1972.

[Wil01] D. C. Wilson. *Case-Based Maintenance: The Husbandry of Experience*. PhD, Indiana University, 2001.

[ZY98] Z. Zang and Q. Yang. Towards Lifetime Maintenance of Case Based Indexes for Continual Case Based Reasoning. In F. Giunchiglia, editor, *Artificial Intelligence: Methodology, Systems, and Applications: 8th International Conference, AIMSA-1998*, volume 1480 of *LNCS*, pages 489–500, Sozopol, Bulgaria, 1998. Springer.

[ZY99] J. Zhu and Q. Yang. Remembering to Add: Competence-Preserving Case Addition Policies for Case Base Maintenance. In T. Dean, editor, *Procs.*

of the 16th International Joint Conference on Artificial Intelligence, IJCAI–99, volume 1450, pages 234–241, Stockholm, Sweden, 1999. Morgan Kaufmann.