Clustering Methods for Requirements Selection and Optimisation

Varsha Veerappa

University College London

Submitted in fulfilment of the requirements

for the degree of Doctor of Philosophy in Computer Science

University College London

Gower Street, London WC1E 6BT, UK

September 2012

Declaration

I, Varsha Veerappa, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Abstract

Decisions about which features to include in a new system or the next release of an existing one are critical to the success of software products. Such decisions should be informed by the needs of the users and stakeholders. But how can we make such decisions when the number of potential features and the number of individual stakeholders are very large? This problem is particularly important when stakeholders' needs are gathered online through the use of discussion forums and web-based feature request management systems. Existing requirements decision-making techniques are not adequate in this context because they do not scale well to such large numbers of feature requests or stakeholders. This thesis addresses this problem by presenting and evaluating clustering methods to facilitate requirements selection and optimization when requirements preferences are elicited from a very large number of stakeholders. Firstly, it presents a novel method for identifying groups of stakeholders with similar preferences for requirements. It computes the representative preferences for the resulting groups and provides additional insights in trends and divergences in stakeholders' preferences which may be used to aid the decision making process. Secondly, it presents a method to help decision-makers identify key similarities and differences among large sets of optimal design decisions. The benefits of these techniques are demonstrated on two real-life projects - one concerned with selecting features for mobile phones and the other concerned with selecting requirements for a rights and access management system.

Acknowledgement

This dissertation would not have been possible without the guidance and the help of many people. I wish to thank, first and foremost my supervisor Dr. Emmanuel Letier for his most valuable guidance, inspiration and supervision throughout this journey. His words of wisdom, encouragement and support have enabled me to get through the toughest times and overcome all the obstacles that have come in my way. I am also very grateful to Prof. Anthony Finkelstein who always found time for me whenever I needed his advice. His suggestions have been always very inspiring and helped me focus on my work.

I am also very grateful to Dr. Soo Ling Lim for kindly providing me with the RALIC data she has collected during her work and for her assistance and useful ideas. I would also like to thank Dr. Yuanyuan Zhang for providing me the implementations of the NSGA-II search-based algorithms. I am thankful to all my friends from the Computer Science Department at UCL who have at some time or another provided me support and advice during my PhD.

Last but not least, I would like to express my deepest appreciation to my parents and family who, despite the distance, have stood by me at all times. I am particular grateful to my husband Anoop for his support and patience throughout.

Table of Contents

List of Figures

List of Tables

Chapter 1 - Introduction

This chapter introduces the motivation for the thesis, lays out the objectives and main contributions of this work.

Making the right requirements decisions is key to successful software projects (Boehm and Papaccio 1988; Lamsweerde 2009a; Heaven and Letier 2011). However, within the new context of online elicitation of requirements and online gathering of stakeholders' preferences for requirements, making correct requirements decisions has become a very complex exercise. This is because existing techniques do not cope well with the large volume of data that such online systems generate.

This thesis uses clustering techniques to aid the requirements decision making process in the context of large web-based requirements elicitation. We achieve this in two steps. Firstly, we apply clustering algorithms to group stakeholders with similar preferences to form more homogeneous stakeholder groups. Secondly, we aid decision makers in analysing the sets of optimal solutions generated by grouping similar solutions.

1.1 Requirements Engineering

Requirements engineering is the process through which the purpose of a software system is discovered. The stakeholders and their needs, as well as the system constraints and their interactions are identified and documented such that they can be used for further analysis, dissemination and implementation (Nuseibeh and Easterbrook 2000; Zave 1995). In other words, we need to discover, understand, formulate, analyse and agree on what the problem is, why the problem needs to be solved and who should be responsible in solving that problem (Lamsweerde 2009a).

The requirements process is a spiral process and involves four major phases, namely (i) domain understanding and elicitation, (ii) evaluation and negotiation, (iii) specification and documentation and (iv) quality assurance (Lamsweerde 2009a) . The dependencies between these activities in the requirements process are illustrated in Figure 1.

This thesis focuses on activities during evaluation and negotiation. During the *evaluation and negotiation* phase, requirements engineers make informed decisions about the best trade-offs among conflicting objectives of the system to-be (Lamsweerde 2009a). Risks, conflicts and alternatives are analysed to produce a prioritized list of requirements. This process may require negotiation among stakeholders to reach some consensus.

Figure 1 The requirement engineering process (Lamsweerde 2009a)

1.2 Large Scale Requirements Elicitation

Requirements engineers are increasingly eliciting requirements from stakeholders using online tools. The accessibility of such tools enables a large number of stakeholders to get involved in the requirements process. They can readily suggest new requirements or features which they think are needed in the system being investigated. Furthermore, with these new tools, the stakeholders can express their preferences for these requirements or features. These emerging trends in requirements engineering activities mean that requirements engineers end up with a large number of requirements to choose from to include in the next release of the application.

Online requirements elicitation tools are web-based applications such as forums, wikis, and recommender systems which also provide functionalities that help

requirements engineers to further elicit and prioritize requirements from these very large number of stakeholders (Laurent and Cleland-Huang 2009). Such systems help collecting data that are useful for understanding stakeholders' preferences, identifying conflicts, and guiding requirements selection and prioritisation (Lim, Quercia, and Finkelstein 2010). The volume of this data can become overwhelming when we are dealing with large numbers of stakeholders.

Examples of such web-based tools include StakeSource (Lim 2010), OPCI (Castro-Herrera, Cleland-Huang, and Mobasher 2009) and OneDesk[1]. These are online collaborative tools that enable stakeholders to submit requirements and rate them. These tools provide options to help decision makers when selecting a subset of requirements using the ratings provided by the stakeholders and they hold other useful information on the stakeholders which can be used to understand the stakeholders' preferences. For example, StakeSource (Lim 2010) uses the stakeholders' influence on the project and their ratings to prioritize requirements and produce ranked list of requirements. StakeSource also keeps stakeholders' profile information.

1.3 Requirements Selection and Prioritisation

Requirements selection and prioritisation is an important activity in requirements engineering. After the elicitation phases, requirements engineers end up with various possible alternatives for solving the problem at hand. They have to help the project's decision makers to choose one or a subset of alternatives from the list of identified ones to design a system that will meet the objectives of the stakeholders as closely as possible. For example, when designing an online calendar for a university, we may have to choose between sending reminders by email, sms or a desktop alert or a combination of these. Each of these alternatives has a cost and they might be preferred by some stakeholders and rejected by others. In this context, the stakeholders are the students, the academic staff, the technical staff and the administrative staff.

To determine which alternatives to include, requirements engineers use decision-making techniques to identify the best trade-offs among the preferences of multiple stakeholders (Lamsweerde 2009b). These techniques can be either qualitative - they do not involve any numbers to measure the impact of the alternatives or they can be

[1] www.onedesk.com

quantitative where numbers are provided to measure the impact of alternatives on the stakeholders' goals.

Traditional cost-value based requirements prioritisation techniques use the relative costs and value of each requirement for each stakeholders' group to compute a ranked list of requirements. Examples of such approach include the AHP process (Saaty 1980) , Karl Wiegers' Requirements Prioritisation Model (Wiegers 2003), Rational Focal Point[2] and Volere requirements prioritisation technique (Robertson and Robertson 2006).

More recent requirements selection techniques have been looking at requirements prioritisation and selection as a multi-objective decision problem (Zhang 2010). Requirements engineering problems are multi-objective by nature; they must take into consideration stakeholders' goals that cannot be directly compared to each other. In such context, there is generally not one single solution that performs better than all others for all objectives simultaneously. Trade-offs must therefore be made between the different objectives to reach a consensus. Multi-objective decision techniques aim to help decision makers make such trade-offs in an informed way. Figure 2 illustrates the multi-objective requirements and prioritisation process used in (Zhang 2010).

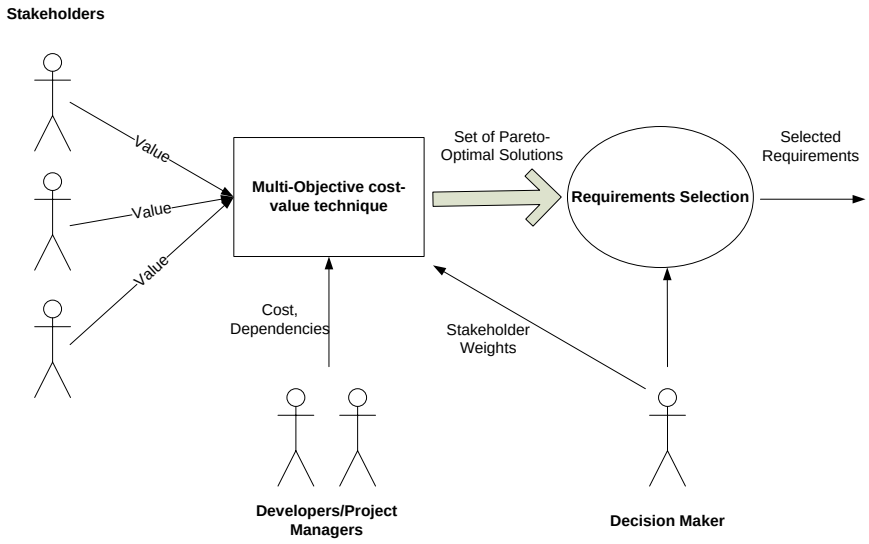


Figure 2 Multi-Objective Requirements Prioritisation and Selection process

The decision maker needs to decide which subset of requirements to implement based on trade-offs on objectives. The process is as follows:

[2] http://www-01.ibm.com/software/awdtools/focalpoint/

1. Stakeholders provide values that represent their preferences for the requirements being considered.
2. Developers or Project Managers working on the project provide their estimation of costs of the requirements. This can be the number of man days or the cost of hardware required for the implementation of the requirements. Where possible, they also provide dependencies that may exist between the requirements, for example, the order in which the requirements need to be implemented.
3. The decision maker may also assign weights to the stakeholders to weigh their values according to their importance in the project.
4. The multi-objective cost-value requirements prioritisation technique used in Figure 2 produces a set of Pareto-optimal solutions from which the decision maker selects a solution to be implemented. A solution is Pareto-optimal if there is no other solution with a higher value and same or lower cost. This set of Pareto-optimal solutions is usually represented on a Pareto Optimal front which plots the solutions according to their objective attainments. Figure 3 shows an example of a Pareto-optimal front.

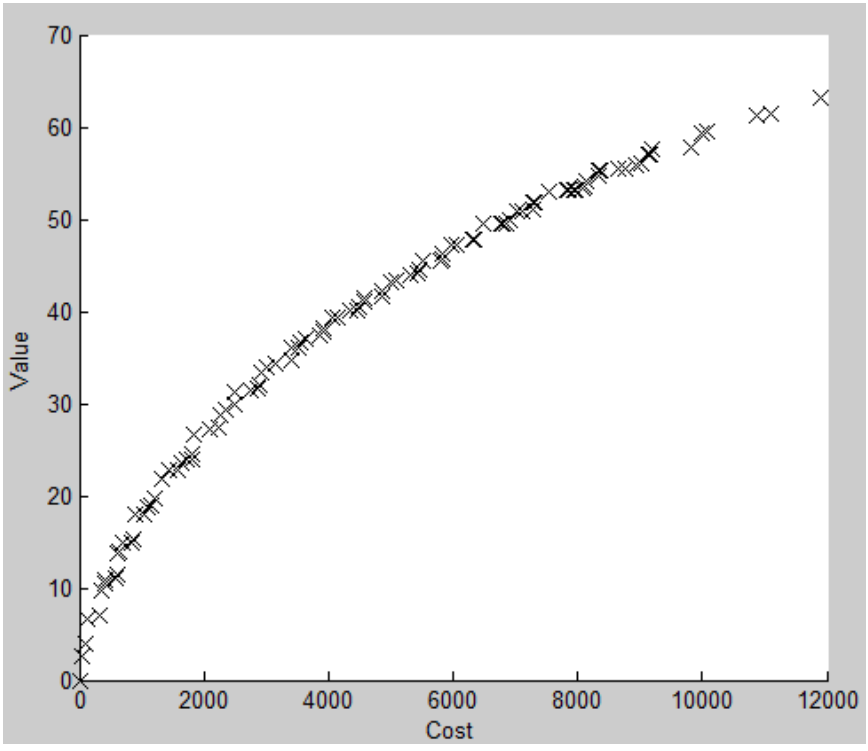


Figure 3 Example Pareto front

Many techniques use multi-objective requirements prioritisation and selection. In the context of system evolution, the problem is known as the Next-Release-Planning problem where value is maximised and cost minimized (Baker et al. 2006). Another important example –found in NASA's Defect Detection Prevention (DDP) framework– is the problem of selecting a set of mitigation actions to reduce the project risks so as to maximize the level of goal attainment while minimizing cost (Feather and Menzies 2002). Fairness analysis is also an extension of multi-objective requirements prioritisation and selection where criteria defining fairness to stakeholders are maximized (Finkelstein et al. 2009).

1.4 Motivation for the Thesis

With the advent of web-based requirements elicitation platforms, the input to the requirements decision-making process has changed. Now, we have hundreds of stakeholders who have rated hundreds of requirements. This implies that decision-making techniques have to process a larger volume of data as we have many more stakeholders and requirements than in the traditional requirements engineering setting.

1.4.1 Limitations when Handling Large Numbers of Stakeholders

Requirement decision-making techniques used for the purpose of requirement prioritisation and selection have been developed in a context where values for a small number of requirements are elicited from a small number of stakeholders or groups. For example, AHP prioritisation (Saaty 1980) performs well when we are prioritising up to 20 requirements (Lauesen 2002). They hardly scale to the context of these online requirements elicitation tools where values and requirements are elicited from a large number of individual stakeholders. Lim defines a large scale project as one having more than 50 stakeholder groups and 10000 stakeholders (Lim 2010). An example of such a project is the FBI Virtual Case File project (Goldstein 2005) where there are 50 stakeholder groups and 12 400 agents who are potential users of the system.

Another example is in the RALIC case study concerned with prioritising requirements for an access control system at University College London (Lim 2010). The project aims at providing identification and access control for all University College London (UCL) buildings on a single card. An initial group of stakeholders involved in the project have been asked to recommend other stakeholders and suggest requirements. All the stakeholders have then been asked to rate the requirements that have been elicited. For

this project, we have 76 stakeholders rating 99 requirements. This dataset has been compiled during the StakeSource project (Lim 2010) and is publicly available online[3]. Another example where such large amount of information related to stakeholders' preferences can be elicited is the Mozilla online feature management system[4] where we have around 1500 active registered users per month.

Furthermore, these techniques assume that homogenous groups of stakeholders can be identified a priori, and that all stakeholders within a group agree on the value to be given to each requirement. Another challenge specific to online elicitation tools is that some groups of stakeholders are likely to be under-represented or over-represented in the collected ratings. For example, stakeholders who have more time to express their preferences online are likely to be over-represented compared to more busy stakeholders whose opinion may be no less important to the project success.

Moreover, ranking the requirements based on a single numerical value in such a context tends to hide conflicts among stakeholders' preferences. If a stakeholder has rated a particular requirement 5 stars and four others have rated it zeros stars, the average and median ratings for that particular requirement will be 1 and 0 stars respectively. Using these figures will not reflect the fact that one stakeholder has given a high value to that particular requirement and that 4 others do not care about it. Requirements engineers may in practice need to be informed of such divergences in ratings to be able to better understand the needs of the stakeholders.

1.4.2 Limitations when Analysing Large Pareto Fronts

The solutions generated by search-based optimization algorithms help decision makers explore trade-offs between the objectives by exposing how much one goal can be improved by compromising on some other goals. For example, in cost-value based requirements prioritisation, such solutions can help explore trade-offs on how much value can be gained by increasing the cost. In large scale problems, these algorithms generate hundreds of solutions. Understanding them and selecting one among them becomes a very complex and time consuming exercise for decision makers. In such problems, a set of solutions with similar cost and value could be composed of either

[3] http://www.cs.ucl.ac.uk/staff/S.Lim/phd/dataset.html

[4] https://wiki.mozilla.org/Main_Page

solutions that are all minor variants of one another (that is, they agree on all major decisions and only differ on smaller, less important ones) or solutions that include major design alternatives (that is, they select significantly different sets of requirements). Currently, such information – which is important to make informed design decisions – is not highlighted to decision makers and finding these manually in large solutions set can be very tedious.

1.5 Clustering Methods to Improve Requirements Decisions

The aim of our work is to improve requirements decisions by using clustering algorithms. This is especially useful to effectively reduce the volume of information when we are looking at large scale problems with large numbers of stakeholders and requirements. We apply clustering techniques at two different stages of the requirements evaluation and selection process.

1.5.1 Clustering Stakeholders

Our technique uses clustering techniques for identifying homogenous groups of stakeholders that can be used as input to existing requirements selection and prioritisation techniques. The technique takes as input the individual stakeholders' preferences for a set of requirements to be evaluated. It produces a set of stakeholder clusters together with their corresponding group preferences for each requirement as output. These group preferences can then be used by existing decision-making techniques to rank the requirements or generate a Pareto front and analyse fairness. A good grouping is one where all the resulting clusters are composed of stakeholders with similar preferences. A group preference for a given requirement is a good representation of its individual stakeholders if it is as close as possible to the preference of all the stakeholders in the group for that requirement.

As a simple example, if a requirement is given very high rating by half the stakeholders and very low by the other half, splitting the stakeholders into two groups with a very high and very low group preference for the requirement for each is better than having a single group where the requirement is given a medium group preference. In the latter case, the group preference fails to represent anyone's preference accurately and the result of decision-making techniques using this value will possibly satisfy no one. When generating groups, there is always a conflict between minimizing the number of groups and maximizing their homogeneity. An extreme situation in which each

stakeholder forms a single group would be very homogenous but would not help decision-making.

Our approach relies on clustering techniques used in market segmentation for product development and marketing (Wedel & Kamakura 1999). In this area, one distinguishes between a customer's characteristics that are product-independent such as his age, location and revenue and characteristics that are product-dependent such as his perceptions, benefits and loyalty for the product. Our approach groups stakeholders based their ratings which are product-dependent characteristics, instead of grouping them according to product-independent characteristics.

1.5.2 Clustering Solutions on the Pareto Front

Here, our objective is to help decision makers make better informed decisions when selecting a particular solution from a set of optimal solutions. Our approach consists of designing a clustering approach with an adequate notion of solutions "closeness" to form useful clusters in requirements selection problems. We cluster solutions based on how close they are to each other in the solution space (that is, based on their similarity in terms of requirements selection) rather than how close they are in the objective space (that is, how close they are in terms of cost and value for example).

We present the generated clusters using specific visualizations to provide requirements engineers with the information they need in such decision processes. We thus help the requirements engineers to find how the cluster composition varies along the Pareto front produced from multi-objective search techniques. This enables them to find how similar or different neighbouring solutions are and make a preliminary selection based on groups of solutions before choosing a final one.

1.6 Contributions of this Thesis

The contributions of this thesis are as follows:

1. Stakeholders' preferences analysis.

   We have developed a technique that clusters stakeholders according to their preference such that a large number of stakeholders participating in web-based requirements elicitation can be reduced to a smaller number of groups of stakeholders that can be used in requirements decision-making techniques. Unlike other techniques where stakeholder groups are found a priori, for example

based on their role in the project, our technique groups stakeholders based on their preference for requirements in the product. We believe this leads to better decision making because the group preferences used to represent the stakeholders in the process are closer to their actual preferences. Requirements engineers are also able to identify "outlier" stakeholders and may further investigate why their preferences are different from others if required. Statistical analyses can also be performed on the clusters to identify preference trends that may exist in them.

2. Optimal Solutions Analysis.

   We have applied clustering algorithms to group solutions on the Pareto Optimal front resulting based on their design similarities. Requirements engineers or decision-makers can thus more easily understand large sets of Pareto-optimal solutions: instead of having to inspect a large number of individual solutions, they can look at a much smaller number of groups of solutions where solutions that belong to a same group are close one to another in terms of selected requirements, and solutions that belong to different groups are significantly different from one another. Requirements decisions can be made incrementally; instead of having to select one solution in a large set of individual solutions, decision makers can first decide for a group of solutions before selecting one solution within the group. Areas on the generated Pareto front where significantly different requirements selections have similar levels of objective attainment are exposed. This may be important for reasoning about system extension and contraction (Parnas 1979).

3. We have developed a tool to implement our techniques. The tool consists of two parts; the first one related to clustering stakeholders. It takes as input the set of requirements ratings provided by the stakeholders and generates the clusters of stakeholders with their representative preference values. The second part of the tool clusters solutions sets that are obtained from multi-objective search techniques. The tool then produces a series of visualisations that show how the clusters vary along the Pareto front as well as what solutions they are composed of.
4. We have evaluated our proposed techniques on the data which has been elicited in conjunction with the UCL RALIC project. We have also evaluated our work

using an industrial dataset obtained from Motorola (Baker et al. 2006). This dataset concerns stakeholders from four mobile telephony service providers which have been asked to rate the features that should be included in handheld communication devices. During the stakeholders' preference analysis, we have been able to find trends in the preferences of stakeholders and improve the preference value use to represent them in the decisions. Clustering solutions during the Pareto analysis has given us insights into the different major sets of solutions for these projects.

1.7 Publications

Work from this thesis has previously been published by the author in the following papers:

- V. Veerappa and E. Letier, *Understanding Clusters of Optimal Solutions in Multi-Objective Decision Problems*, Proceedings RE 2011 – 19th IEEE International Requirements Engineering Conference, Trento, Italy, September 2011.
- V. Veerappa and E. Letier, Clus*tering Stakeholders for Requirements Decision Making*, Proceedings REFSQ 2011 – 17th International Working Conference on Requirements Engineering: Foundation for Software Quality, Essen, Germany, March 2011.

1.8 Thesis Organisation

The rest of the thesis is organised as follows:

Chapter 2 is the background section describing the various quantitative decision-making techniques that are used in requirements engineering and cluster analysis.

Chapter 3 presents a general overview how our framework for decision optimisation with large numbers of stakeholders and requirements.

Chapter 4 presents our technique for clustering stakeholders based on their preferences.

Chapter 5 presents our technique for clustering solutions on the Pareto Optimal front generated from multi-objective methods.

Chapter 6 evaluates our clustering techniques on two large case studies.

Chapter 7 concludes the thesis and presents limitations and future work.

Chapter 2 - Background

This chapter reviews concepts needed to understand the techniques developed in the thesis.

In Chapter 1, we have introduced how our work applies clustering techniques to cluster large numbers of stakeholders. Where multi-objective search-based requirements prioritisation and selection techniques have been used, our technique clusters similar solutions on the Pareto front. In this section we will be looking at the relevant background required to understand the technical aspects of work. We first cover the concepts behind requirements prioritisation and selection techniques relevant in our context. We then look at clustering algorithms with particular attention to hierarchical clustering algorithms.

2.1 Eliciting Stakeholders' Preferences

Over the recent years, there has been an increasing trend towards the elicitation of requirements and preferences over the web using online tools such as wikis, forums, recommender systems and other social platforms. These types of projects often involve large number of requirements and stakeholders from all walks of life, located in various part of the globe (Laurent and Cleland-Huang 2009). Information that has been collected using these platforms, not only enable requirements engineers to become aware of the requirements of stakeholders but also provide other useful information which can be used to understand the stakeholders preferences during the prioritisation process. For example, context information such as location or browser type can help understand why some stakeholders prefer some features over others.

One such online tool is StakeSource (Lim 2010) which is a web-based crowdsourcing platform that enables stakeholders to recommend other stakeholders, suggest requirements and rate requirements. An initial set of stakeholders is identified for a project and they are asked to recommend other stakeholders that they think have a say in the project. The initial stakeholders are assigned weights based on how important they are with respect to their "networks" of recommended stakeholders. The stakeholders are also provided with the possibility to suggest requirements for the project and rate their preferences for existing requirements. The requirements engineer can use the tool to

further prioritise the stakeholders and requirements that have been elicited. Between December 2009 and December 2010, the tool has been used in 10 industrial projects in countries like the UK, Japan, Australia and Canada in both software and non-software related projects (Lim et al. 2012). These projects involved from 10 to 200 stakeholders.

OPCI (Castro-Herrera, Cleland-Huang, and Mobasher 2009) is also an example where requirements are gathered using web-based platform. This kind of platform collects large volumes of data concerning requirements, all of these contributed by the stakeholders who use the forum. OPCI then groups the stakeholders according to the contents of their posts to profile them based on their interests. Another example of an online crowdsourcing platform is the iRequire (Seyff, Ollmann, and Bortenschlager 2011) that is built on the Samsung bada platform. iRequire enables stakeholders to suggest features or requirements for mobile applications (apps) on the move using their mobile phones.

In addition to these platforms issued from research, commercial tools providing similar functionalities have started to emerge. Examples of such platforms include OneDesk[5] and IBM Rational RequisitePro[6].

Some of these platforms already support rating of requirements/features by stakeholders (e.g. StakeSource) and the general trend is towards such online rating systems by assigning stars (similar to Amazon[7]) or more explicitly assigning a preference value on a scale. We next describe the type of scales usually used in the context of preference assignment to requirements.

2.1.1 Measurement of Preference

Rating scales are the most common means for people to indicate their preference for a given option in surveys and ratings systems. These rating scales can be numerical interval or text-based ordinal ones. The numerical scales most commonly used include the one to ten or one to five scales where people are asked to choose from option to represent how strongly they feel about a particular statement. For example, on web-

[5] http://www.onedesk.com/

[6] http://www-01.ibm.com/software/awdtools/reqpro/

[7] http://www.amazon.co.uk/

based platforms such as Amazon[8], users are provided with a five-star rating system where they assign one to five stars to some item to indicate how much they like it or how much it is useful to them.

Text-based ordinal rating scales are mostly Likert scales. The Likert scale(Likert 1932) is a psychometric measure that is often used to measure attitude of respondents with respect to some product or feature. This measure has both direction and magnitude that is, it shows whether someone has positive or negative attitude and how much of it (Raden 1985). For example, customers may not be satisfied with a service with the same intensity. The standard Likert scale will consist of Likert items which are the individual options in the rating scale. For example a five scale Likert scale may have the following items: *strongly disagree*, *disagree*, *neither agree nor disagree*, *agree*, *strongly agree*. Likert scales may be used to measure agreement, frequency, quality and likelihood.

2.1.2 Representative Value for Stakeholder Preference

When we have a large number of stakeholders, we may need to compute a single representative value for their preference ratings that can be used in the decision making process. If qualitative scales like the Likert Scale have been used, we may have to encode the values into a numeric scale before further processing .The main assumption here is that the numerical scale which the Likert Scale has been converted to is an interval one – a numerical scale where the distance between two items are equal (Siegel 1957)- on which we can do further statistical analysis.

Statisticians use specific techniques to determine a representative value, called measures central tendency, for large volumes of data. Measures of central tendency commonly used in statistics include the mean, the median and the mode (Crawshaw and Chambers 2001).

Similarly, in our context, when large numbers of ratings have been collected from stakeholders, a value representative of the preference of all stakeholders is usually determined for each of the requirements. Decision makers then use this value as input in any subsequent computations to make decisions. This representative value can be one of the measures of central tendency used in statistics.

[8] http://www.amazon.co.uk/

We have chosen the median as representative value of the preferences for the stakeholder clusters our technique discovers as it is not affected by ratings with extreme values on the scale making it a more realistic representation of the middle value of the data (Crawshaw and Chambers 2001). However, if the rating scale is interval in nature and there are no extreme values, the decision maker may still opt for the mean.

2.2 Search-Based Software Engineering

Software engineering involves mostly optimisation problems. Software engineers are continuously finding the best tradeoffs between resources and functionalities at the various phases of the software life cycle. Examples of such tradeoffs include choosing the right allocation of resources to software projects or minimizing the set of test cases such that all of the branches of a program are covered during the testing phase. Such problems are well tailored for search-based software engineering.

Search-based software engineering (SBSE) (Harman 2007; Harman and Jones 2001) transforms traditional software engineering problems as search-based optimisation problems. Such problems are those which look for optimal or near optimal solutions in a search space of candidate solutions following a fitness function that differentiates between better and worse solutions in that space. A wide range of optimisation and search techniques have been used for this purpose. The most common ones are local search, simulated annealing, genetic algorithms and genetic programming(Harman, Mansouri, and Zhang 2009).

SBSE has been applied to a range of problems at different stages in the software engineering lifecycle(Harman, Mansouri, and Zhang 2009), including maintenance (Mancoridis et al. 1999; Mancoridis et al. 1998; Reformat and Miller 2003; Reformat, Chai, and Miller 2007),testing (Yoo and Harman 2007), verification (Alba and Chicano 2007), design (Räihä, Koskimies, and Mäkinen 2008; Ma and Zhang 2008), and requirements engineering (Zhang 2010).

The work in this thesis contributes to the area of search-based software engineering for requirements problems, as detailed in the following section.

2.3 Multi-Objective search-based Requirements Prioritisation and Selection

We next explain the concepts behind multi-objective requirements selection and optimization and explain what a Pareto front is.

If we have two objectives, cost and value, we can formulate the requirements selection problem as follows.

Let $R = \{r_1, r_2, \dots, r_n\}$ be a set of n requirements where each requirement r_i has a cost c_i that denotes how much it costs to implement it and a value $v_i$that denotes its value to the project's stakeholders. We can represent cost and value as vectors c and v respectively:

$$c = [c_1, c_2, \dots, c_n]$$

$$v = [v_1, v_2, \dots, v_n]$$

The aim of the decision maker is to select some subset of requirements in R. We represent this solution by a vector $x = [x_1 \ x_2 \ \dots \ x_n]$ where x_i equals 1 if requirement r_i is selected and equals 0 otherwise. For example, $S_1 = [1 \ 1 \ 0 \ 0 \ 1 \ 0 \ ]$ represents a solution where r_1, r_2 and r_5 are selected and the others are excluded.

The total value and total cost of a solution x are defined as follows:

$$value(x) = \sum_{i=1}^{n} v_i \times x_i \quad (4)$$

$$cost(x) = \sum_{i=1}^{n} c_i \times x_i \quad (5)$$

Our objectives when selecting a solution are to maximize value and minimise cost:

$$Maximize\ (value)$$

$$Minimize\ (cost)$$

One way of choosing an appropriate solution is to use multi-objective search techniques. Our work relies on Zhang's technique (Zhang 2010) which uses the Non-Dominated Sorting Genetic Algorithm-II algorithm (NSGA-II) (Deb et al. 2002) to perform multi-objective search-based requirements selection and prioritisation

The NSGA-II is derived from the concept of dominance(Deb 2001). In this search algorithm, two solutions are compared on the basis of whether one dominates the other solution or not in terms of objectives attainment. The search uses evolution for this purpose. It starts with an initial population of solutions and chooses the fittest solutions among it. This is determined by using a fitness function that calculates the level to which the solution attains the objective of the search and selects the solutions that are the fittest. In our case, these are solutions with maximum value and minimum cost. It then forms a new population (offspring) from these fit solutions by merging parts from them (parents) and mutating the new solutions. The process is repeated until a stopping condition is reached. In this thesis, we perform 20 runs of the algorithm for 50 generations on a population size of 200. The algorithm is summarised in Figure 4. The result is a *non-dominated* set of solutions which the algorithm discovers up to that stopping condition.

The fitness function is key to the success of the NSGA-II algorithm. Given that we have M fitness functions $f_i(x)$ where $i = 1,2,3 \dots M$, then our aim is to find solutions that will optimise those fitness functions. Thus, a solution x_1 is said to dominate a solution x_2 iff the following conditions hold (Zhang 2010):

$$f_i(x_1) \geq f_i(x_2) \quad \forall\, i \in \{1, 2, 3 \dots, M\} \text{ and}$$

$$\exists i \in \{1, 2, 3, \dots, M\} \mid f_i(x_1) > f_i(x_2)$$

1. Randomly generate or seed initial population *P*
2. Repeat until no. of runs completed
3. Evaluate fitness of each solution in *P*
4. Select parents from *P* according to selection mechanism
5. Recombine parents to form new offspring
6. Construct new population P' from parents and offspring
7. Mutate P'
8. $P \leftarrow P'$

Figure 4 Genetic algorithm description (McMinn 2004)

Plotting the Pareto-optimality for the solutions gives a Pareto front on which the set Pareto-optimal solutions in the solution space lie. Figure 5 is an example of such a Pareto front for the objective value and cost.

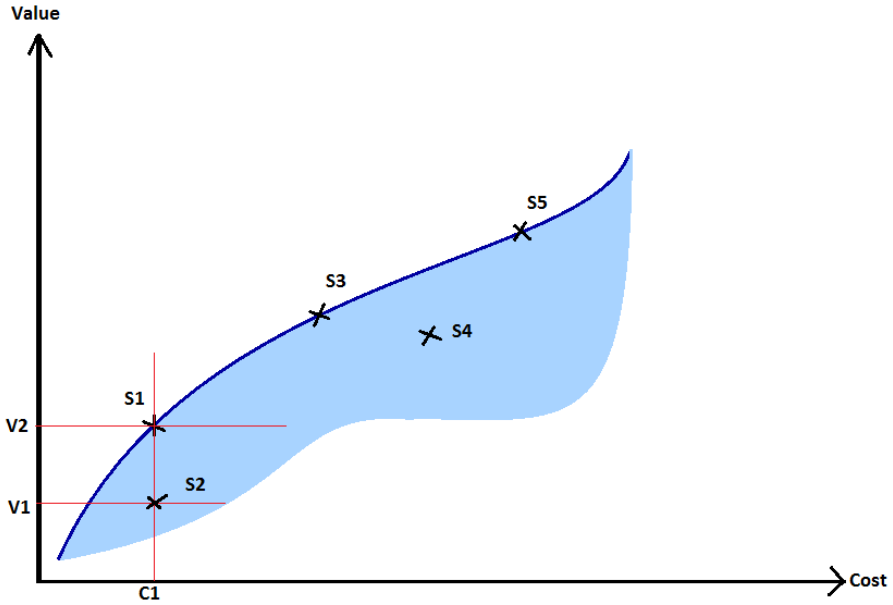


Figure 5 Pareto Optimal Front

In Figure 5, solutions S1, S2, S3, S4 and S5 are sets of requirements represented by vectors. Solution S1 *dominates* solution S2 as for the same cost C1, it has a higher value V2. Similarly, solution S4 is dominated by solution S3 as it has higher value for a lower cost. Thus, S1, S3 and S5 are Pareto-optimal solutions.

Multi-objective search-based requirements selection and optimisation techniques have many advantages (Zhang, Finkelstein, and Harman 2008). These techniques are robust as they take constraints and changes into consideration when looking for near-optimal solutions and thus these remain near-optimal even under change. They also enable decision-makers to make sensitivity analysis on the solutions as each solution is a variation of the objectives' attainment. The Pareto optimal front generated from these techniques gives insights such as "elbows" which show areas on the Pareto front where considerable changes in cost/value ratio occur. As shown in Figure 6, at around the area circled is an elbow and beyond this, any increase in cost does not significantly improve the value.

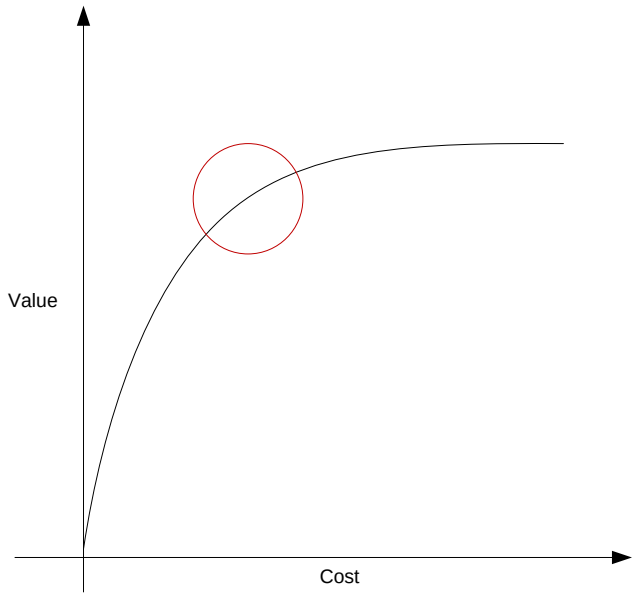


Figure 6 Example of an elbow on a Pareto front

Search-based requirements prioritisation and selection techniques are however not without limitations (Zhang, Finkelstein, and Harman 2008). The major issue remains scalability of the algorithms. An increase in the number of requirements, stakeholders, constraints and complexity inevitably causes an increase in the resources and time required to execute the algorithms. Pareto fronts on more than 3 objectives are difficult to represent graphically. There is also a significant lack of tools and techniques to support the analysis of the solutions to the stakeholders. Technical aspects such as selection of the appropriate algorithm and a good fitness function can be tedious and is often a trial and error exercise. Finally, more work needs to be done to incorporate concepts such as requirements dependencies and partial fulfilment of "continuous" requirements. For the purpose of our work, we will assume that dependencies issues have been catered for in the search algorithm.

2.4 Post-Pareto Analysis

Multi-objective search techniques produce a set of Pareto Optimal solutions (Zhang 2010). The next step for decision makers is to choose one single solution from this set of solution for implementation. However, if the set of Pareto Optimal solution is too large, analysing the different solutions becomes a complex and time consuming exercise. This step is crucial for the success of the project and can be enhanced with techniques that highlight information pertinent to the decisions to be made. Such techniques include

visual depictions of the Pareto Optimal front or more thorough analysis of the solutions to reduce the solution space to choose from.

2.4.1 Visual Analysis of Objectives Attainment

These types of approaches display the different objectives attainment of the non-dominated solutions using graphs. Where we have up to three objectives, it is quite straightforward to plot these objectives and visually compare them. However, this becomes impossible with more than three objectives. One way of analysing the different non-dominated solutions in this case is the use of self-organizing maps (Obayashi and Sasaki 2003) which is a neural network model that maps high number of dimensions in data to two dimensions represented by neurons. Heat maps can also be used to analyse the set optimal solutions (Pryke, Mostaghim, and Nazemi 2007). The solutions are clustered together based on their objective attainments and these are plotted on a heat map where columns and rows represent the parameters and the objectives. The shade of a cell in the map depicts the value of the parameter or objective for the solution.

2.4.2 Solutions Reduction Analysis

In cases where a large number of solutions are present in the Pareto Optimal front, one way of facilitating post-Pareto analysis is by comparing the objective attainment of these solutions and find groups of solutions with similar objective attainment in the optimal set of solutions. Techniques used to achieve this include clustering analysis (Morse 1980; Rosenman and Gero 1985; Mattson, Mullur, and Messac 2004). The decision maker then decides which one of these representative solutions he wants to implement.

Our approach is different from those two as we are focusing on grouping solutions by design similarity and not objective attainment.

2.5 Clustering techniques

Data clustering is at the very core of this thesis. We use this technique in Chapter 4 to group similar stakeholders together while in Chapter 5 we use it to group similar solutions on the Pareto front. Key concepts here are *similarity measures*, *cluster quality/validity measures* and the *clustering algorithms* themselves.

Data clustering is the method of grouping objects into clusters such that objects in one cluster are similar to each other while objects from different clusters are quite distinct. Here, there is no prior knowledge about the possible structure of the groups. Data clustering is also known as clustering analysis, segmentation analysis, taxonomy analysis or unsupervised classification (Gan, Ma, and Wu 2007). This is not to be confused with supervised classification where objects are assigned to groups already identified beforehand. Objects in our context are either stakeholders or solutions.

Clustering techniques can be classified into three main categories, these are non-overlapping, overlapping and fuzzy techniques (Wedel and Kamakura 1999) . In non-overlapping clustering methods, when objects are assigned to a group, they will belong only to that group. Non-overlapping clustering methods can be further broken down into hierarchical and non-hierarchical methods. Hierarchical methods proceed by identifying hierarchical relations among the objects based on some measure of how similar their attributes and behaviours are. Non-hierarchical methods, on the other hand, group objects directly based on the similarity in the raw data.

2.5.1 Hierarchical Clustering Algorithms

There are two main types of hierarchical clustering methods; these are agglomerative and divisive methods. Agglomerative methods start with single-element clusters and these are merged based on their similarity until a single cluster is obtained. Divisive methods on the other hand, start with a single cluster with all elements in it. This single cluster is split repeatedly based on the dissimilarity of its elements until single-element clusters are obtained. The similarity measures used in each case can be obtained from variables measured on the elements. What to compare usually depends on the choice of the decision makers doing the cluster analysis. Agglomerative clustering algorithms are more widely used in practice than divisive ones. Unlike divisive algorithms, agglomerative algorithms do not misclassify objects with rare attributes and are thus more likely to produce correct clusters (Everitt, Landau, and Leese 2001).

A hierarchical clustering method is a procedure to transform a proximity matrix - a matrix which represents the distance between all pairs of objects in a dataset - into a sequence of nested partitions (Jain and Dubes 1988). This can be described as follows. Let

$O = \{o_1, o_2, \dots, o_n\}$ denote a set of n objects to be clustered.

A partition L of O is a set $\{G_1, G_2, \ldots, G_m\}$, where each G_i is a set of objects such that $G_i \cap G_j = \emptyset \quad for\ i\ and\ j\ from\ 1\ to\ m, i \neq j$ and $G_1 \cup G_2 \cup \ldots \cup G_m = O$

Clusters are the components of a given partition. A partition Q is nested in the partition L if every component in Q is a subset of a component of partition L. For example, if L and Q are partitions of the set of objects $\{o_1, o_2, \ldots, o_{10}\}$ and

$$L = \{(o_1, o_2, o_6, o_9, o_{10}), (o_3, o_4, o_5, o_7, o_8)\}$$

$$Q = \{(o_1, o_2), (o_6, o_9, o_{10}), (o_3, o_4, o_5, o_7), (o_8)\}$$

then we can say that partition Q is nested into partition L.

No consensus has been reached on whether hierarchical clustering algorithms (both agglomerative and divisive) fare better than other clustering algorithms (Cutting et al. 1992; Larsen and Aone 1999; Zhao and Karypis 2002; Jain and Dubes 1988). Hierarchical algorithms have no backtracking capability and a poor time complexity but they are deterministic and hence more predictable than other types of clustering algorithms such as K-means (Hartigan and Wong 1979). Furthermore, the nested nature of the partitions allows different users to choose different partitions, according to the desired similarity level. These algorithms are preferred when we do not know the number of clusters or the structure of the data beforehand as it is the case in our context.

The techniques presented in this thesis use *hierarchical agglomerative clustering algorithms* to cluster data in our work.

Dendrograms

A dendrogram is a mathematical and graphical representation of the different steps involved in the hierarchical clustering process. Figure 7 is an example of a dendrogram consisting of 7 objects O_1 to O_7. The leaves at the bottom of the dendrogram are individual objects which are uniformly spaced on the horizontal axis. The vertical axis represents the distance or dissimilarity measure between the individual objects or clusters of objects. A dendrogram consists of nodes which represent points where clusters are joined together to form bigger clusters and stems which represent the distance at which the clusters are joined (Everitt, Landau, and Leese 2001). The dendrogram enables us to visualise how the number of clusters and cluster composition vary with distance. Cutting off a dendrogram at certain distance gives clusters at every

point where the stems of the dendrogram are crossed. For example, a cut-off at a distance 3 on Figure 7 gives 5 clusters as the line y=3 crosses the dendrogram at 5 places.

The dendrogram is a key feature of hierarchical clustering algorithms as it provides important insights to the person performing the clustering process. For example, outliers can easily be identified from dendrograms. Outliers will be joined to other clusters in the dendrogram at a very late stage. In Figure 7, object O_7 seems to be an outlier as it is individually merged with the rest of the clusters at a very high distance. Another useful information conveyed by dendrograms is the validity of clusters formed. For example in Figure 7, we can clearly see that objects O_5 and O_6 form a good cluster as this cluster remains as a single cluster until very late in the clustering process.

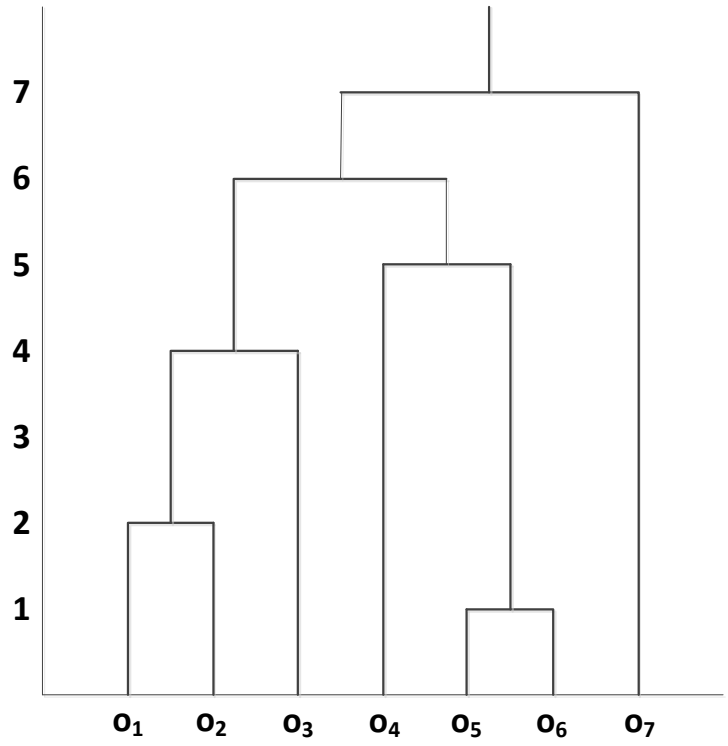


Figure 7 Example of a dendrogram

Cophenetic Correlation Coefficient

Hierarchical clustering algorithms force a particular structure on the data. This structure can be acceptable or it can distort the actual relationships among the objects (Everitt, Landau, and Leese 2001). The Cophenetic Correlation coefficient (Sokal and Rohlf 1962) determines how well a dendrogram replicates the naturally occurring structures in the data.

The Cophenetic Correlation coefficient is the linear correlation coefficient between the Cophenetic distances obtained from the Cophenetic Matrix of the dendrogram, and the original distances used to construct the dendrogram. Thus, it is a measure of how faithfully the dendrogram represents the distance among objects. The Cophenetic matrix represents the heights h_{ij} where two objects i and j (clusters or individual objects) are merged on the dendrogram (Everitt, Landau, and Leese 2001). A Cophenetic Coefficient value closer to 1 represents a better clustering for the data. This metric can be used to choose between two clustering algorithms to apply on a dataset. If we have a Cophenetic Matrix Z and the original distance matrix Y for a dataset, the Cophenetic Correlation coefficient, is given by

$$c = \frac{\sum_{i<j}(Y_{ij} - y)(Z_{ij} - z)}{\sqrt{\sum_{i<j}(Y_{ij} - y)^2 \sum_{i<j}(Z_{ij} - z)^2}}$$

where:

- Y_{ij} is the original distance between objects *i* and *j*.
- Z_{ij} is the Cophenetic distance between objects *i* and *j*
- *y* and *z* are the average of Y and Z respectively.

The Cophenetic Correlation coefficient is used to choose the best hierarchical clustering algorithm to use for the data being investigated.

Stopping rules

At some point during the cluster analysis process, the user will have to determine which groups generated by the clustering algorithm he wants to use by specifying a cut-off value on the dendrogram. One commonly used rule of the thumb here is to look at large changes at which the objects merge on the dendrogram. These generally indicate well-formed clusters below that level (Everitt, Landau, and Leese 2001).

More formal methods have been proposed to assist users in this process. These have varying complexities depending on the way they compute the optimal number of clusters or cut-off values. One such method is the Mojena's cut-off value which we describe next.

Mojena's cut-off value

Mojena has proposed a particularly appropriate measure known as the Mojena's cut-off value to determine the best cut-off value for a dataset based on the structure of the dendrogram (Mojena 1977). This heuristic has the advantage of being quick and provides fairly good groups (Milligan and Cooper 1985). If there are N objects in the dataset, the Mojena's cut-off is determined from the heights of the stems in the dendrogram where objects and clusters are merged to form the N-1 possible clusters for the dataset. The Mojena's cut-off value M is computed as follows:

$$M = \tilde{h} + \alpha S_h$$

where: $\check{h}$ is the average of the dendrogram heights for all N−1 clusters.

S_h is the standard deviation of the dendrogram heights for all N−1 clusters.

α is a constant that has been chosen empirically to yield good results.

Milligan and Cooper recommend a value of 1.25 for α (Milligan and Cooper 1985). The Mojena's cut-off value is used in this thesis to provide an initial quick indication of what the recommended cut-off should be in the dataset being analysed.

2.5.2 Similarity Measures

Clustering algorithms use two ways levels of similarity measures to group objects. The first one is the intra-cluster proximity measure which measures how two objects within a cluster are similar or close. The second one is the inter-cluster distance which measures similarity between two clusters.

Intra-cluster Similarity Measures

Intra-cluster measures depend on the nature of the data and the problem being investigated. For example, for numerical data types, the most common similarity measure is the Euclidean distance. Other similarity measures include the Jaccard distance (Jaccard 1908), City block distance, Gower's distance (Gower 1971) among others. These distance measures can be tuned to fit the requirements of the clustering exercise at hand. For example, a weight can be assigned to specific attributes when computing the proximity measure calculation to give them more importance. We refer to this measure as the distance between the objects in this thesis.

Inter-cluster Similarity Measures

In agglomerative hierarchical techniques, the similarity between clusters is commonly measured by five measures of distance: the single linkage, the complete linkage, the average linkage, the centroid linkage and the median linkage. Figure 8 illustrates how these linkages are computed.

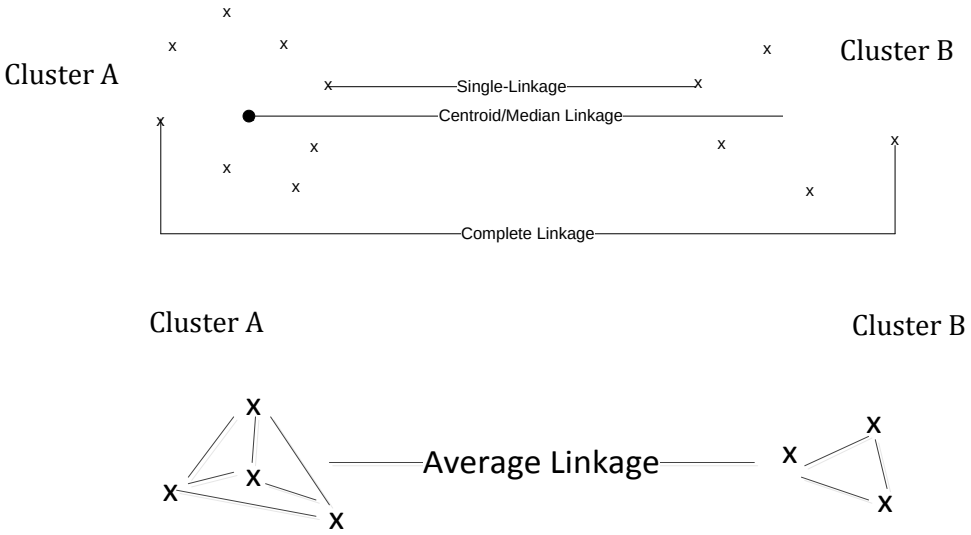


Figure 8 Inter-cluster Similarity Measures

Let us assume we have two clusters A and B. The single linkage (Sneath 1957) between A and B is the shortest distance between any two members in them; the complete linkage (Sørensen 1948) is the longest distance between any two members in them; the average linkage (Sokal and Michener 1958) is the average distance among all pairs of objects in A and B. A variant of the average linkage is the weighted average linkage (McQuitty 1966) which weighs the average distances based on the size of the clusters. This is particularly useful when we have uneven cluster size. Another widely used inter-cluster similarity measure is the Ward's method (Ward 1963) which measures the sums of square of the distance within clusters.

Other inter-cluster distance measures include the median linkage (Gower 1967) and the centroid linkage (Sokal and Michener 1958) that use the actual objects in the clusters rather than the distance between them to compute distances between clusters. The centroid linkage (Sokal and Michener 1958) is the distance between the centroids – mean

vector of all objects in a cluster- of the two clusters. The median linkage is similar to the centroid linkage except for the additional step of weighing the mean vectors based on the size of the clusters.

An important consideration when choosing which of these similarity measures to use is the type of data being analysed. For example for the single, average and complete linkages, the distance between two objects is enough as input but for centroid, Ward's and median linkages, the actual objects must be provided and it must be possible to compute an Euclidean distance between two of these objects.

Each of these measures has advantages and disadvantages. The single linkages and complete linkages for example are less computational intensive, but the single-linkage is prone to chaining and reversal problems (Everitt, Landau, and Leese 2001). Chaining happens when distinct clusters are forced to join together because of "noise" objects while reversal is observed when later clusters are joined together at distances smaller than that at which earlier clusters were merged. Average linkage takes into account cluster structure unlike complete linkage but the latter is less likely to be affected by observational errors and is very widely used. Ward's method on the other hand, tends to impose a spherical structure to the clusters formed.

In Chapters 4 and 5, we use the complete, the average and weighted average linkages.

2.5.3 Cluster Quality

To evaluate the cluster quality, we need to measure how compact and well separated the generated clusters are. Cluster quality can be determined from two characteristics of the cluster. The first characteristic to consider is the distance among the objects also known as *cluster cohesion*. A cluster will be valid if the distance among the objects in that cluster is minimised. The other characteristic is the distance between the clusters also known as *separation*. Valid clusters will have the distance between them maximised. Another way of assessing if a cluster is of good quality is by determining how genuine the group is. This can be achieved by comparing which objects are always grouped together when different clustering algorithms are used. Figure 9 and Figure 10 illustrate examples of a good and a bad clustering respectively.

Existing measures of cluster quality check for either internal distance, external distance or both. These measures are dependent on the type of data being classified.

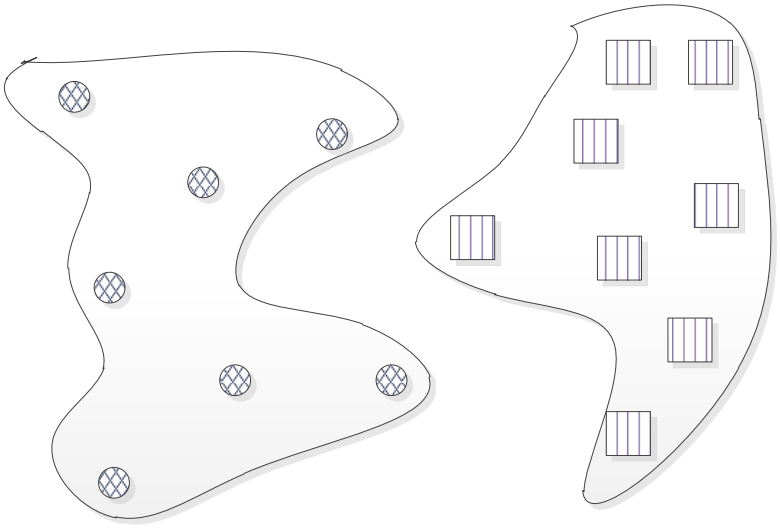


Figure 9 A good clustering

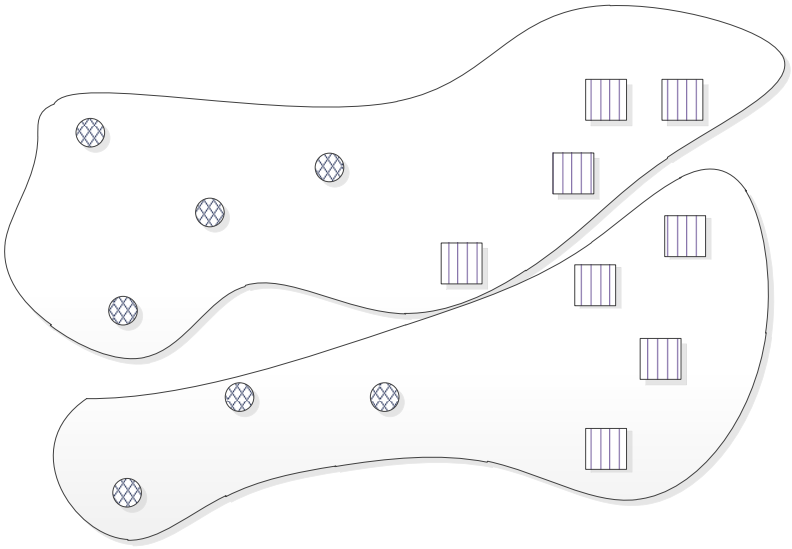


Figure 10 A bad clustering

C-index

The C-index (Hubert and Levin 1976) is an example of a measure that assesses the internal quality of clusters.

For a given cluster, it is computed as follows:

$$C = \frac{S - S_{min}}{S_{max} - S_{min}}$$

where S is is the sum of the distances between all pairs of objects in the cluster, and if n is the number of such pairs

S_{min} is the sum of the n smallest distances of all pairs within the full dataset

S_{max} is is the sum of the n largest distance of all pairs within the full dataset

S_{min} and S_{max} correspond to the smallest possible and largest possible sum of distances between the objects for all possible clusters of size n within the dataset. A good cluster is one where S is close to S_{min}, and therefore C close to 0.

The overall C index for a partition is computed as the average of the C indices obtained for each cluster in that partition.

Silhouette Index

The Silhouette index (Rousseeuw 1987) measures the cluster cohesion and separation for each cluster. It is computed as follows:

$$S_i = \frac{(b_i - a_i)}{\max(a_i, b_i)}$$

Where a_i is the average distance of object i with respect to all other objects in the same cluster.

b_i is the minimum of the average distances of object i with respect to all objects in the next closest cluster.

The Silhouette index can take values between -1 and 1 inclusive. An index of 1 means a well formed cluster while an index of -1 indicates a bad clustering. The overall Silhouette index of a partition is the overall average of Silhouette indices for the objects. One drawback of the silhouette index is that it does not perform well in instances where we have outliers. In these cases, a misleading Silhouette index value 1 is obtained. This will inflate the overall average Silhouette index value and give the impression of a good partition even if that is not the case. When this happens, Rousseeuw recommends setting

the Silhouette index value for that outlier single item cluster to 0 before computing the average Silhouette index value (Rousseeuw 1987).

The Rand Index

The Rand index (Rand 1971) is an efficient heuristic to determine if the clusters formed at a particular cut-off represent the natural groups in the data as closely as possible. This method compares the groups formed by two different clustering algorithms and determines the proportion of objects that are in the same cluster and the proportion of objects in different clusters for the both algorithms. The Rand index takes values between 0 and 1 where 0 indicates no object fall in the same clusters and 1 indicates all the clusters agree exactly. Thus for a given set of n objects, the Rand Index, R, is computed as follows:

$$R = \frac{a+b}{\binom{n}{2}}$$

where a is the number of pairs of objects in the same cluster in both clusterings.

b is the number of pairs of objects in different clusters in both clusterings.

The Adjusted Rand Index is a variant of the Rand Index that accounts for chance when measuring the proportions and has been recommended for use by Milligan and Cooper (Milligan and Cooper 1985).

In this thesis, we refer to the Adjusted Rand index as the Rand index.

2.6 Summary

This section reviewed theories behind large-scale requirements elicitation and multi-objective requirements prioritisation and selection. We have also looked at cluster analysis and related concepts fundamental in understanding the techniques that fall under clustering. We focused on hierarchical clustering algorithms with explanation about related notions specific to them.

Chapter 3 - Framework Overview

This chapter explains how our proposed techniques work

Our approach consists of applying clustering methods to improve the requirements decision making process at two specific stages of the decision making process. For the first aspect of our technique, we are working in the context where we have a large number of stakeholders who have provided their preference ratings for requirements using large scale requirements elicitation tools resulting in a large volume of data to be used as input decision making techniques. The second aspect of our technique considers the output of decision making techniques that generate Pareto Optimal fronts when a large volume of data has been used as input. Figure 11 gives an overview of our framework.

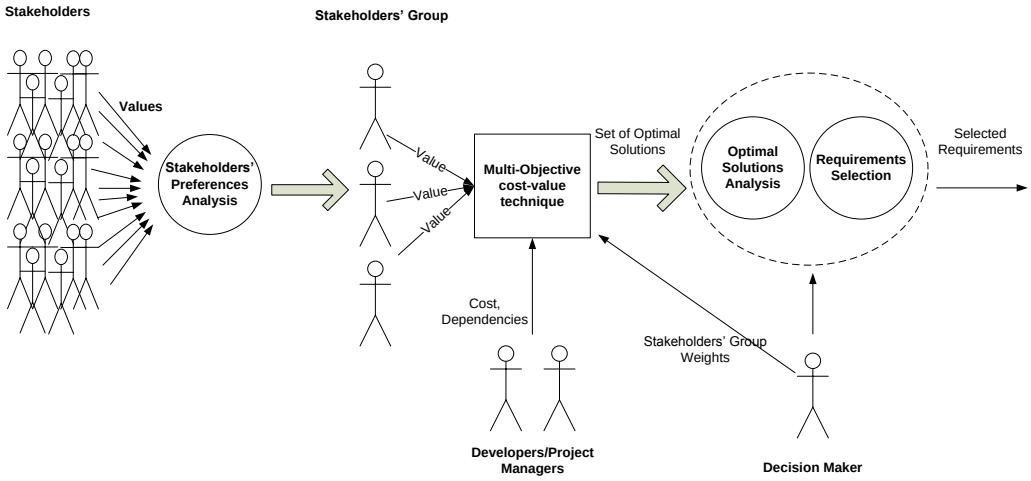


Figure 11 Clustering stakeholders and solutions in large-scale requirements selection

3.1 Stakeholders' Preference Analysis

First, we apply clustering algorithms when ratings have been elicited online from a large number of stakeholders. We call this technique the *stakeholders' preferences analysis.* Here, we generate groups of stakeholders with similar preferences and recommend a preference value to represent this group in the decision process.

The typical scenario for the application of our stakeholder preferences analysis technique is as follows:

1. A large number of stakeholders provide their preferences via large-scale web-based requirements elicitation tools for requirements to be included in the next release of the application.
2. These preference ratings are fed to the *stakeholders' preferences analysis* technique as input.
3. The *stakeholders' preferences analysis* technique produces clusters of stakeholders with similar preferences and computes cluster preference values for each of the clusters.
4. The decision maker may further analyse the composition of the groups for trends if he needs to.

The *stakeholders' preferences analysis technique* determines cluster preferences that are as close as possible to the individual ratings each stakeholder has provided initially. Using these cluster preferences in the decision making techniques ensures that the stakeholder preferences are better represented in the decisions.

Our technique decreases the volume of data being fed to the decision making technique as many stakeholders are reduced to a few groups of stakeholders with similar preferences. Once these groups have been formed, it is also possible to identify preference trends and exceptions that are present among the stakeholders. These can lead to further investigation that may enable the requirements engineers to better understand the diversity in the stakeholders' preferences.

3.2 Optimal Solutions Analysis

Where multi-objective search-based requirements prioritisation techniques have been used, we also use clustering techniques to cluster solutions on the Pareto Optimal front. We refer to this technique as the *optimal solutions analysis.* Our technique finds groups of solutions with similar designs and provides the decision maker with visualizations and statistical analysis to understand the groups. The decision-maker then selects an optimal solution based on the analysis made.

If the requirements selection is made based on cost and value, the typical scenario for the *optimal solutions analysis* technique in this case will be as follows:

1. The multi-objective search-based requirements prioritisation technique will use the costs and values for each requirement to produce a set of optimal solutions.

2. The *optimal solutions analysis* technique uses clustering to group solutions with similar selection of requirements.
3. It also provides graphical displays of the composition of the clusters. These include a plot of the Pareto optimal front which provides a visual display of how the clusters of solutions are distributed on it.

This means that once our technique has been applied to the Pareto Optimal front, the decision maker can locate a set of solutions that he can focus on and perform further analysis to determine how the designs within the set changes. This can help him determine how solutions with very similar value and cost vary in terms of selected requirements. Another benefit our technique is that decision makers can choose solutions incrementally – first choose a group of similar solutions from a reasonable number of groups of similar solutions and then choose a particular solution from the selected group.

Figure 12 illustrates how our framework can be applied to projects.

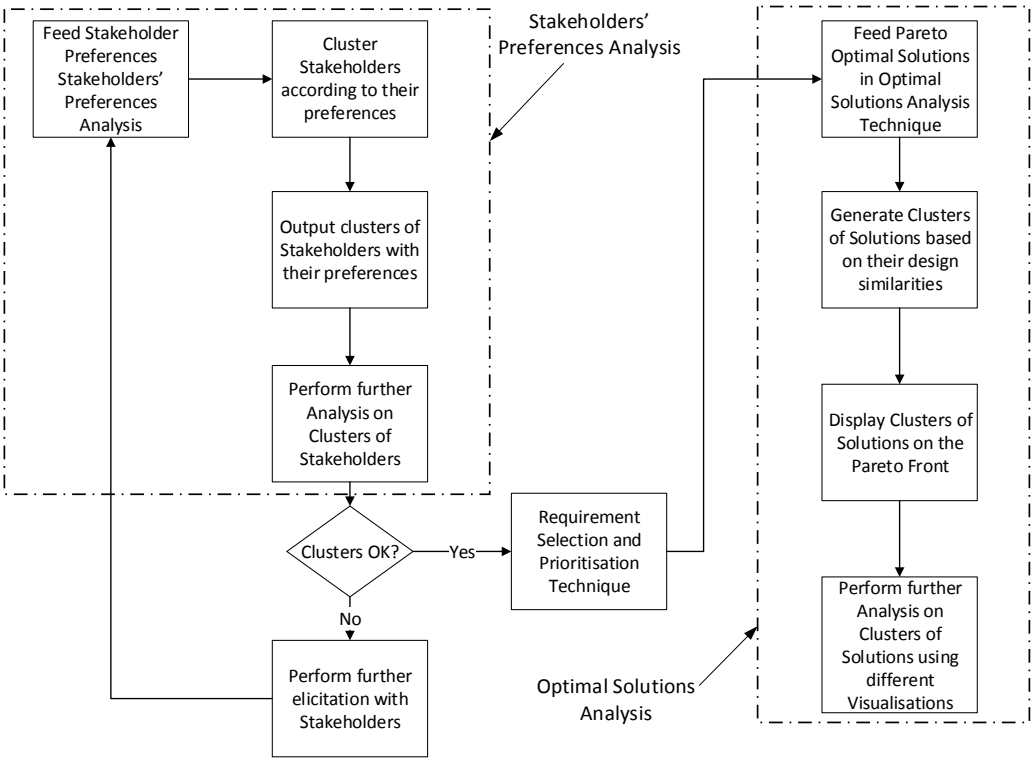


Figure 12 Application of proposed framework

3.3 Terminology

We next provide a definition of the terminology used in the rest of this thesis. Figure 13 shows how these different terms are related to each other.

- Requirement

  A requirement is a feature to be enforced by the software-to-be alone or with conjunction with other system components (Lamsweerde 2009a) .

- Stakeholder

  A stakeholder is an individual who have an interest in the system and who may influence the way the system is designed (Lamsweerde 2009a).

- Preference

  The preference of a stakeholder for a requirement is extent to which that stakeholder believes the requirement is useful to him.

- Stakeholder Group

  In our context a stakeholder group is a group of stakeholders with similar preferences.

- Solution

  A solution is combination of requirements that can be implemented in the system.

- Cost

  The cost of a requirement is the cost of implementing this single requirement in the system.

  The total cost of a solution is the sum of the costs of all the requirements included in the solution.

- Value

  Value quantifies the preference of a stakeholder for a requirement. In our context, it is a number on a given scale e.g. -1 to 5.

The total value of a solution is the sum of the values for the stakeholders for the requirements included in the solution.

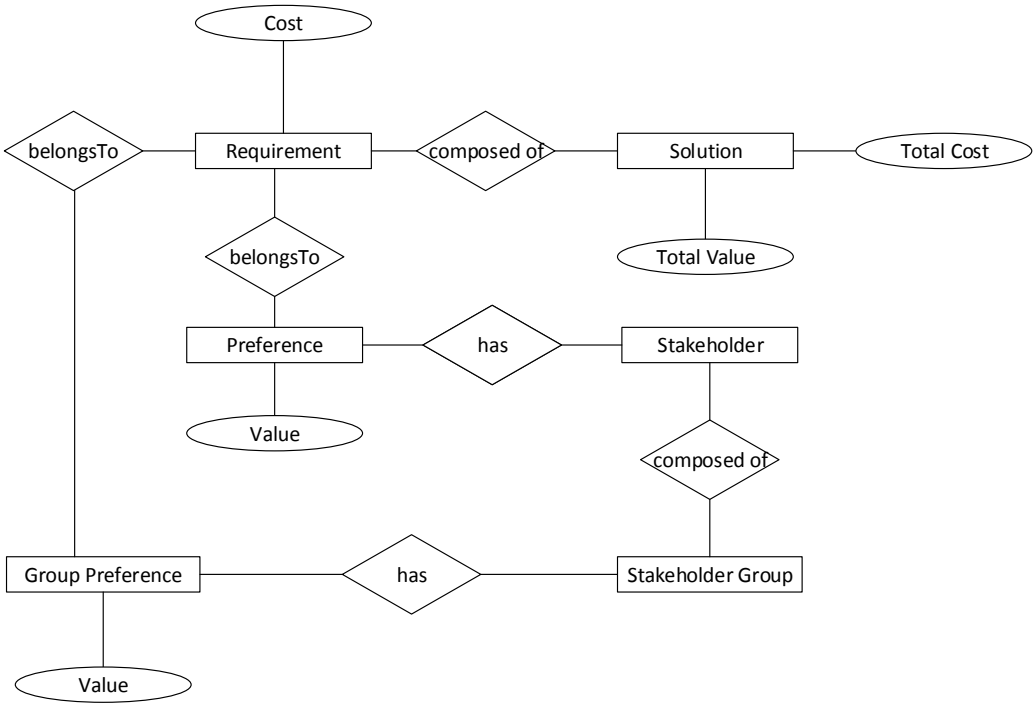


Figure 13 Relationship among terms

Chapter 4 - Stakeholders' Preferences Analysis

We apply clustering algorithms to group stakeholders by preference

4.1 Introduction

In this chapter, we present our technique to cluster stakeholders according to their preferences in the context where we have a large number of stakeholders and requirements. As highlighted in Figure 14, this part of our work, the *stakeholders' preferences analysis*, produces the input to be fed into requirements prioritisation techniques. We introduce the context in which our technique is useful and elaborate on concepts that we use in devising our technique. We use a small artificial dataset to illustrate related concepts as we introduce them. Validation of this technique on real data will be described in Chapter 6.

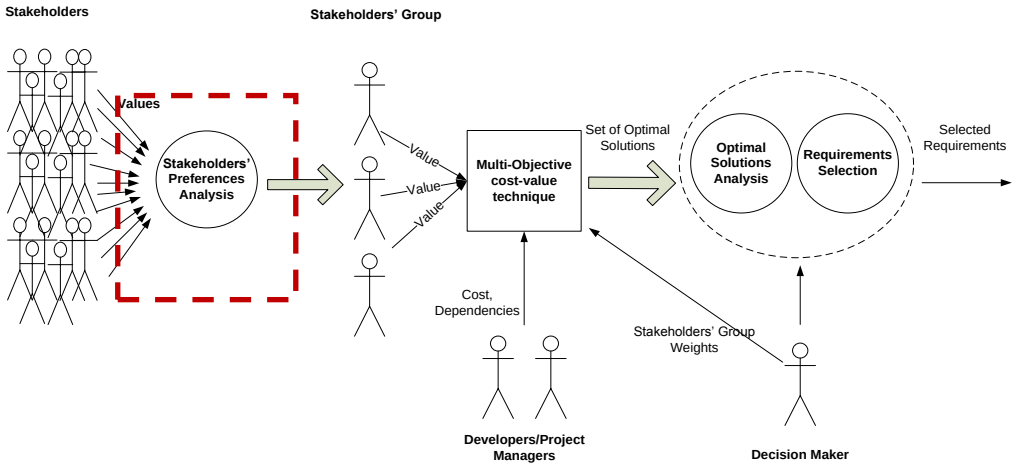


Figure 14 Scope of stakeholders' preferences analysis

As seen earlier in Chapter 1, existing requirements prioritisation techniques do not scale well as the number of requirements and stakeholders increases. The aim of the stakeholders' *preferences analysis* is to use clustering techniques to help decision makers understand this scale of data. This scenario will typically arise in systems where large scale requirements elicitation is done.

We have implemented a tool in Matlab to enable requirements decision makers to use our technique and make necessary analysis to understand the clusters of stakeholders that our technique discovers.

4.2 Grouping Stakeholders by Preference

This *stakeholders' preferences analysis* borrows concepts from market segmentation theory (Wedel and Kamakura 1999) from the field of marketing. Marketing and Requirements engineering have common aims in their respective fields.

In the field of marketing, professionals gather information about customers with respect to the products they want to market. They often ask people to answer to surveys or rate features/products they like. They also gather other demographic data such as age and location along in this process. They then perform market segmentation on the data collected to build homogenous groups of customers with common interests from a large pool of heterogeneous customers. This creates groups of customers that have similar responses to the market mix (Wedel and Kamakura 1999). The process of forming and analysing the customer groups is known as market segmentation. Similarly, in our context, requirements engineers ask stakeholders to rate requirements of the system to be. This is very similar to the preference elicitation being carried out in marketing. We can therefore use market segmentation techniques similar to those used by marketing professionals to understand the preference of stakeholders for these features.

Market segments are characterised by bases which are attributes that describe the customers that fall within it. Bases include observable general bases, observable product bases, unobservable general bases and unobservable product bases (Frank, Massy, and Wind 1972). In our context, we are working on the unobservable product specific bases of the system to be. These consist of product-benefits perceptions and importance, brand attitudes, preferences and behavioural intentions which are closely related to preference for features in a product.

Marketing professionals identify these segments using a number of methods(Wedel and Kamakura 1999). This can be achieved either "a-priori" or "post-hoc", that is, before or after consumer data has been collected. The techniques used to group the consumers can further be divided as descriptive or predictive ones. A descriptive technique analyses data across a single segment base and tries to find associations among them while a predictive technique analyses the relation between two set of variables – one is the set of independent base variables and the other is a set of dependent base variables that are influenced by the latter set.

The *stakeholders' preferences analysis* technique forms segments or groups of stakeholders by grouping stakeholders with common preferences together. In this case, we are considering only the preference of stakeholders (after they have done the rating) and we are trying to discover associations among them. This implies that we need to specifically use a post-hoc descriptive technique. Thus, we need to use cluster analysis in our technique as it is the practice for this kind of grouping in the field of market segmentation.

4.2.1 Assumptions

One main assumption is that there are enough stakeholders for our technique to produce meaningful clusters. We also assume that in cases where qualitative scales have been used, the ratings can be directly converted to an interval scale to be able to do analyses on them. This type of conversion is very common in practice (Blaikie 2003). It is acceptable as long as we are aware of its shortcomings (Stevens 1951; Mccall 2001) as the relative differences between items of the ordinal scale is subjective while the differences on the interval scale are in terms of fixed units. Any subsequent parametric test may not always reflect the actual characteristics of the data.

4.2.2 Clustering Stakeholders

We use clustering algorithms, more specifically the hierarchical agglomerative clustering algorithm described in Chapter 2, to cluster our stakeholders based on similarities in their ratings for requirements and hence, similarities in their preferences.

Our technique consists of the following steps:

1. Apply the agglomerative hierarchical clustering algorithm to the stakeholders' ratings for the requirements.
2. Identify the clusters of stakeholders.
3. Determine representative values for requirements for each cluster of stakeholders.

The output from our technique enables the use of statistical or graphical methods for further analysis of the compositions and trends in the clusters of stakeholders.

Running example

To illustrate the concepts underlying our technique we use a fictitious dataset consisting of 9 stakeholders A to I, each rating 5 requirements R1 to R5 on scale of -1 to 5. In this case, a rating of 5 means that the requirement is highly desirable while a rating of -1 means that the stakeholder does not want the requirement in the system. The data for the running example is listed in Table 1 Running Example data. The stakeholders belong to stakeholder groups G1, G2 and G3.

In this thesis, there are to two types of stakeholder groupings. The first one is product independent; it is referred to as the *stakeholder group.* It corresponds to groupings of stakeholders based on their characteristics such as their age, location or role in the system. The other one is product dependent; is referred to as the *stakeholder cluster*. It corresponds to groupings of stakeholders based on their preference for requirements of the system. So G1 to G3 in this case are product independent groups that can be stakeholder groupings based on their location for example.

In the running example, Stakeholder A has low values for the requirements R1 to R4 and high value for R5. Stakeholder B has medium value for R1 and R4, high value for R2 and R3 and negative value for R5 (meaning that Stakeholder B does not want R5 to be included in the system). Stakeholder I on the other hand has low values for all of R1 , R3 andR4 and no ratings for R2.

For the ratings that can be compared between A and B, B has high ratings for requirements for which A has low ratings and does not want one of the requirements for which A has high ratings. Thus, these stakeholders' preferences are far from each other. The ratings of Stakeholder C diverge only slightly on R2 and R4 and have the same value as Stakeholder A for the other requirements. This indicates that Stakeholder A and Stakeholder C have preferences which are close to each other. Both of Stakeholder B and Stakeholder D do not want to have requirement R5 implemented

Stakeholders F, G, H and I also have missing ratings from requirements R2 and R4. We discuss later in this chapter how we can compare these stakeholders with missing ratings using the Gower's Distance.

Stakeholder	Stakeholder Group	R1	R2	R3	R4	R5
A	**G1**	1	2	1	2	5
B	**G1**	3	4	5	3	-1
C	**G2**	1	1	1	1	5
D	**G3**	4	5	4	5	-1
E	**G3**	1	1	2	1	3
F	**G1**	3		1	3	3
G	**G2**	4		3	2	3
H	**G2**	2	1	5		5
I	**G2**	1		2	1	4
	Cost	23	76	43	87	64

Table 1 Running Example data

The most important steps when applying the agglomerative hierarchical clustering algorithm to the ratings are the selection of appropriate distance and inter-cluster similarity measures. We discuss these next.

4.2.3 Considerations when Comparing Stakeholders' Preferences

Two aspects of the data have to be considered when comparing the stakeholders' preferences. These are missing data and negative values. The distance measure that we use to compare the stakeholders' preferences must be able to handle any special cases that may arise from these aspects.

Missing data

One common occurrence in rating systems is missing data. Lim also observed the occurrence of missing data when eliciting ratings from stakeholders (Lim 2010). This is often because the stakeholders omit to vote for one or more of the requirements. When we have this kind of data we have two options:

1. Replace the missing values with the neutral "don't care" value on the rating scale and then compute the distance. Very often this will be 0.
2. Use a distance measure that identifies missing data and ignores comparison in those cases.

Before stakeholder clusters have been formed, we want our stakeholders' *preferences analysis* technique to follow option 2. We therefore need to define a stakeholder distance measure that can handle missing data. Once the stakeholder clusters have been formed, a

cluster preference for each requirement can be defined as the median value over all available stakeholders' preferences for that cluster. If there is no rating at all for a requirement within a cluster, we will give the decision makers the possibility to follow option 1 and give the requirement the "don't care" value for that cluster.

Negative values

Some rating systems like the one used for the RALIC case study use a negative value on the scale, for example -1, to show that the stakeholder does not want to have a given requirement in the system. Our distance measure must be able to identify these and handle them as special cases. We discuss how this is achieved later in this chapter.

4.2.4 Distance Measure for Stakeholders' Preferences

It is important to emphasize on the importance of a proper distance measure for a good clustering to take place. The distance measure determines closeness of the stakeholders both within clusters and among clusters. As mentioned in section 4.2.3, the *stakeholders' preferences analysis* technique needs to use a distance measure that can perform even if there are special cases in the data.

One such distance measure the Gower's distance measure (Gower 1971). This distance measure proceeds by first identifying missing data in the dataset and then flags them to be ignored during the computation of the distance. The Gower's distance can be extended to handle negative preferences differently. To illustrate how Gower's distance works, we start by explaining a simple version of it which is the Manhattan distance.

We first lay out the convention we use in this thesis to represent the stakeholders' preferences. If there are m stakeholders rating n requirements, the preference matrix P is defined as follows:

$$\mathrm{P} = \begin{bmatrix} p_{11} & \cdots & p_{1n} \\ \vdots & \ddots & \vdots \\ p_{m1} & \cdots & p_{mn} \end{bmatrix}$$

Where p_{ij} is the preference of stakeholder i for the requirement j.

Manhattan distance

The Manhattan distance also known as the city-block distance is a very simple way of measuring distance between two objects (Black 2006). The Manhattan distance is the

absolute distance between two objects. Thus, the Manhattan distance between two stakeholders a and b is given by

$$D_m(a,b) = \sum_{j=1}^{n} |p_{aj} - p_{bj}|$$

In our context, the objects will be the set of ratings obtained from stakeholders for all of the requirements. Thus the Manhattan distance between the two Stakeholders A and B in the running example is computed as follows:

$$D_m(A,E) = \sum_{j=1}^{5} |p_{Aj} - p_{Ej}|$$

$$D_m(A,E) = |1-1| + |2-1| + |1-2| + |2-1| + |5-3| = 5$$

Similarly, the Manhattan distance between the two Stakeholders A and C is as follows:

$$D_m(A,C) = |1-1| + |2-1| + |1-1| + |2-1| + |5-5| = 2$$

This confirms that the preference of stakeholder A is closer to that of stakeholder C than to that of stakeholder E. This is a very simple way of measuring the similarities among the preferences of stakeholders. However, it can happen that there are missing data or omissions in the preference ratings.

Gower's Distance

The Gower's distance is obtained by calculating the General Similarity Coefficient of Gower (Gower 1971) between two objects. Gower recognises that when comparison is being made between two objects, it will often happen that an attribute may be absent in the objects being compared. This may be because the attribute simply does not exist or because this information is missing in one of the objects. To be able to make comparisons when such cases arise, Gower proposes the General Similarity Coefficient. Thus, if we have two stakeholders a and b, the simplest form of the General Similarity Coefficient of Gower for these two stakeholders is given by:

$$s(a,b) = \frac{1}{n}\sum_{j=1}^{n} 1 - \left(\frac{|p_{aj} - p_{bj}|}{R_j}\right)$$

where $R_j = \max(p_{ij}) - \min(p_{ij})$ and p_{ij} is the preference of stakeholder i for requirement j.

The General Similarity Coefficient of Gower is a normalised form of the Manhattan distance. It will have a value between 0 and 1 for a requirement. A value of 0 means that preference of the stakeholders for that requirement is completely different. A value of 1 on the other hand means that the stakeholders have the same preference for that requirement.

The Gower's distance (Gower 1971) is given by:

$$D_G(a,b) = \sqrt{1 - s(a,b)}$$

The Gower's distance will be a value in the range of 0 to 1 where 0 means that there is no difference between objects being investigated and 1 means that there is absolute difference between them.

Thus the Gower's distance between the ratings of Stakeholders A and E is computed as:

$$D_G(A,E) = \sqrt{1 - \frac{1}{5}\sum_{j=1}^{5} 1 - \left(\frac{|p_{Aj} - p_{Ej}|}{R_j}\right)}$$

which is:

$$D_G(A,E)$$

$$= \sqrt{1 - \frac{1}{5}\left[\left(1 - \frac{|1-1|}{3}\right) + \left(1 - \frac{|2-1|}{4}\right) + \left(1 - \frac{|1-2|}{4}\right) + \left(1 - \frac{|2-1|}{4}\right) + \left(1 - \frac{|5-3|}{6}\right)\right]}$$

$$= 0.47$$

Similarly, the distance between the ratings of Stakeholders A and C is computed as:

$$D_G(A,C)$$

$$= \sqrt{1 - \frac{1}{5}\left[\left(1 - \frac{|1-1|}{3}\right) + \left(1 - \frac{|2-1|}{4}\right) + \left(1 - \frac{|1-1|}{4}\right) + \left(1 - \frac{|2-1|}{4}\right) + \left(1 - \frac{|5-5|}{6}\right)\right]}$$

$$= 0.32$$

Since 0.47 is closer to 1 than 0.32 is, it can be concluded that the preference of Stakeholder A is closer to that of Stakeholder C than to that of Stakeholder E.

Handling Missing Values

To handle missing values, Gower introduces a variable $w_j(a,b)$which is a weight factor that is set to 1 to indicate a comparison between stakeholders a and b is possible for requirement j and to 0 to indicate that no comparison is possible. Thus, the coefficient becomes

$$S_G(a,b) = \frac{\sum_{j=1}^{n} w_j(a,b) s_j(a,b)}{\sum_{j=1}^{n} w_j(a,b)}$$

Thus $w_j(a,b)$ is weight set to 1 when a preference value is present for both stakeholders a and b for requirement j and to 0 when the preference for requirement j is either missing or not available for one or both of the stakeholders. For numerical values, $s_j(a,b)$ is given by:

$$s_j(a,b) = 1 - \left(\frac{|p_{aj} - p_{bj}|}{R_j}\right)$$

where $R_j = \max(p_{ij}) - \min(p_{ij})$

In this case, the Gower's distance is given by:

$$D_G(a,b) = \sqrt{1 - S_G(a,b)} \quad (1)$$

To illustrate how the Gower's Distance accounts for missing values in the dataset, let us compute the distances between Stakeholder G and Stakeholder I.

In this case, equation (1) is used to compute the Gower's distance as both stakeholders' preferences have missing data. The General Similarity Coefficient of Gower is computed first. It then used compute the Gower's distance between Stakeholders G and I as follows:

$$D_G(G,I) = \sqrt{1 - S_G(G,I)}$$

where

$$S_G(G,I) = \frac{\sum_{j=1}^{5} w_j(G,I) s_j(G,I)}{\sum_{j=1}^{5} w_j(G,I)}$$

Since there are no preference values for R2 for stakeholders G and I, $w_2(G,I)$ will have value 0.

$$S_G(G,I) = \frac{1 \times s_1(G,I) + 0 \times s_2(G,I) + 1 \times s_3(G,I) + 1 \times s_4(G,I) + 1 \times s_5(G,I)}{1+0+1+1+1}$$

$$S_G(G,I) = \frac{s_1(G,I) + s_3(G,I) + s_4(G,I) + s_5(G,I)}{4}$$

And

$$s_j(G,I) = 1 - \left(\frac{|p_{Gj} - p_{Ij}|}{R_j}\right)$$

Therefore:

$$S_G(G,I) = \frac{\left(1 - \left(\frac{|4-1|}{3}\right)\right) + \left(1 - \left(\frac{|3-2|}{4}\right)\right) + \left(1 - \left(\frac{|2-1|}{4}\right)\right) + \left(1 - \left(\frac{|3-4|}{6}\right)\right)}{4}$$

$$= 0.58$$

$$D_G(G,I) = \sqrt{1 - 0.58} = 0.65$$

Similarly the distance between Stakeholders G and H is:

$$D_G(G,H) = \sqrt{1 - 0.5} = 0.71$$

Although there is missing data, the Gower's distance has been able to determine that the preference of Stakeholder I is closer to that of Stakeholder G than to that of Stakeholder H.

Handling Negative Values

In order to compute the Gower's distance, we need to compute the value of s_j for each requirement j in the data set. When we compare the preferences of two stakeholders for requirement j, if we have negative preferences for one stakeholder and a positive value for the other stakeholder, we set the value of s_j to 1 to indicate complete dissimilarity

between their preferences. However, if we have negative preferences for both stakeholders, the value of s_j is automatically computed as 0 by the algorithm and this reflects the fact that the stakeholders do not want to have the requirement included.

This is illustrated by comparing the preferences of Stakeholder A with that of Stakeholder B and Stakeholder D. The Gower's distance between stakeholders A and B is given by:

$$D_G(A,B) = \sqrt{1 - S_G(A,B)}$$

$$S_G(A,B) = \frac{s_1(A,B) + s_2(A,B) + s_3(A,B) + s_4(A,B) + s_5(A,B)}{5}$$

In this case, since R5 has a negative value for stakeholder B only, the value of $s_5(A,B)$ is set to 0.

$$S_G(A,B) = \frac{s_1(A,B) + s_2(A,B) + s_3(A,B) + s_4(A,B) + 0}{5}$$

$$D_G(A,B) = \sqrt{1 - \frac{1}{5}\left[\left(1 - \frac{|1-3|}{3}\right) + \left(1 - \frac{|2-4|}{4}\right) + \left(1 - \frac{|1-5|}{4}\right) + \left(1 - \frac{|2-3|}{4}\right) + 0\right]}$$
$$= 0.83$$

The Gower's distance between stakeholders D and B is given by:

$$D_G(D,B) = \sqrt{1 - S_G(D,B)}$$

$$S_G(D,B) = \frac{s_1(D,B) + s_2(D,B) + s_3(D,B) + s_4(D,B) + s_5(D,B)}{5}$$

In this case, R5 has a negative value for both stakeholders B and D.

$$D_G(D,B)$$
$$= \sqrt{1 - \frac{1}{5}\left[\left(1 - \frac{|3-4|}{3}\right) + \left(1 - \frac{|4-5|}{4}\right) + \left(1 - \frac{|5-4|}{4}\right) + \left(1 - \frac{|3-5|}{4}\right) + \left(1 - \frac{|-1+1|}{6}\right)\right]}$$
$$= 0.52$$

4.2.5 Inter-Cluster Similarity Measure

The three candidate inter-cluster similarity measures for the *stakeholders' preferences analysis* technique are the average, weighted-average linkage and the complete linkage algorithms. As described earlier in Chapter 2, these three have the most consistent performance and are more likely to produce more reliable clusterings in our context.

The *stakeholders' preferences analysis* technique uses all three inter-cluster similarity measures and determines the clusterings in each case. It then computes the Cophenetic Correlation coefficient for each of them and determines which measure gives the higher value for the coefficient. It then uses the clusterings with the higher Cophenetic Correlation coefficient for the final clusters.

4.2.6 Generating Clusters of Stakeholders

The hierarchical agglomerative algorithm clustering algorithm used by the *stakeholders' preferences analysis* technique is a bottom up approach. It starts with all stakeholders as individual clusters. Then it merges stakeholders with similar preferences into clusters in a sequential way until all the stakeholders are in a single cluster. This process is depicted on a dendrogram.

The *stakeholders' preferences analysis* starts by computing the Cophenetic Correlation coefficient to determine which inter-cluster distance measure to use. In the case of the running example, the resulting values are:

$Ccc_{Average\ Linkage}$: 0.9347

$Ccc_{Weighted\ Average\ Linkage}$: 0.9231

$Ccc_{Complete\ Linkage}$: 0.9270

Since the Cophenetic Correlation coefficient is larger for the average linkage, it is used to form the clusters. Figure 13 illustrates the resulting dendrogram. The x-axis labels on the dendrogram identify the stakeholders in the order that they are fed to the clustering algorithm.

4.2.7 Identifying the Clusters of Stakeholders

Once the dendrogram has been generated, it is cut off horizontally at a given height to get the clusters that exist at that height. At every point where the cut-off line crosses the dendrogram we have a cluster.

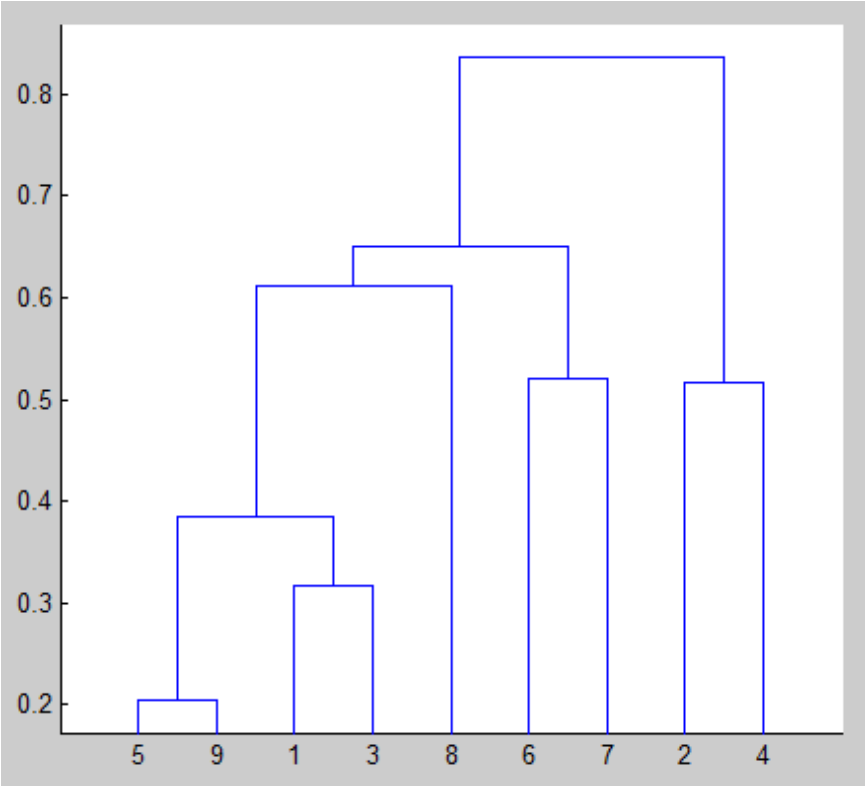


Figure 15 Dendrogram for running example

Our technique helps the decision maker by providing a "default" cut-off value which is the Mojena's cut-off value (described in Chapter 2). He can either choose to use this default value or manually find a cut-off value that better suits him.

For the running example, the default cut-off is at a height of 0.76 on the dendrogram which gives two clusters. However, the decision maker may choose to go for a cut-off at 0.55 to give four clusters. The cut-offs are illustrated in Figure 14.

To further assist the decision-maker, our technique provides the Rand, C and Silhouette indices as shown in Figure 15 to measure cluster quality. In the running example, the C index at 4 clusters is lower than that at 2 clusters. The Rand and Silhouette indices for both cut-offs are very similar. This means that choosing 4 clusters results in better groups of stakeholders in this case.

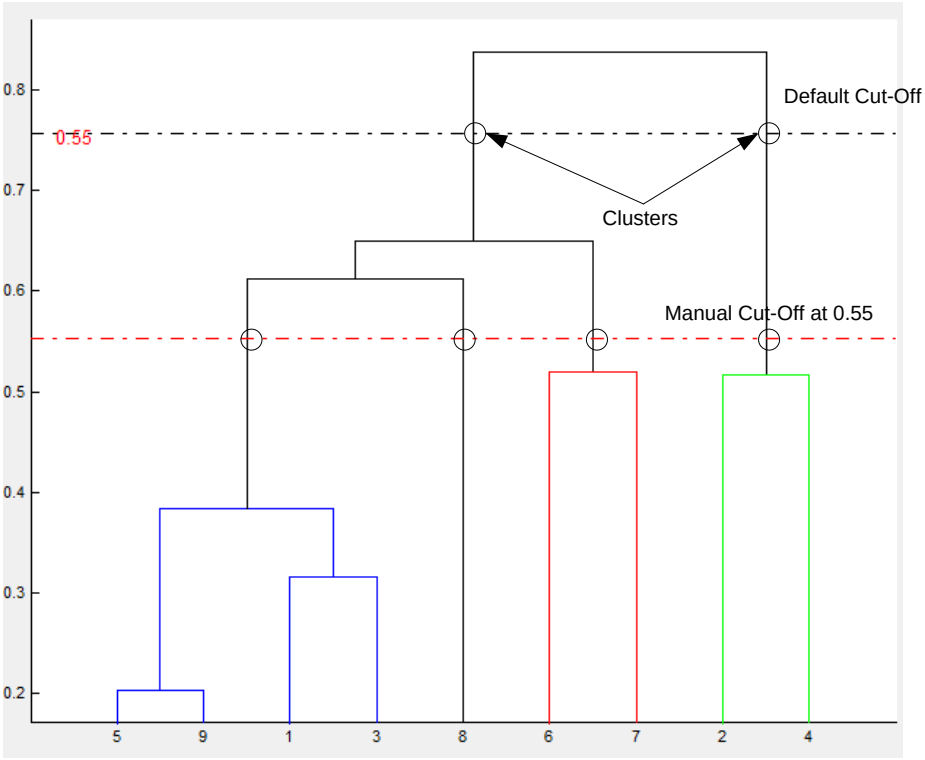


Figure 16 Cut-offs for running example

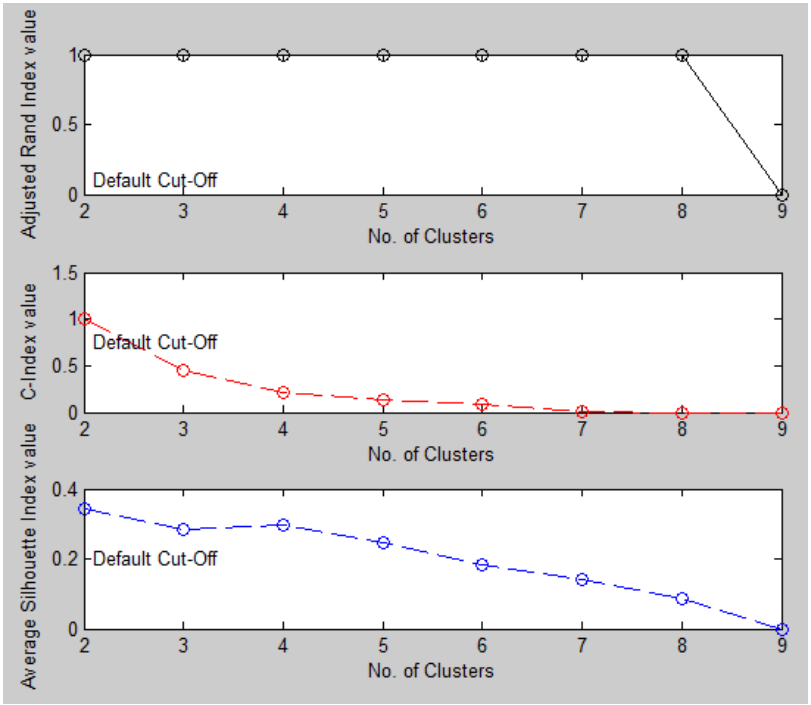


Figure 17 Rand, C, Silhouette index values for running example

Determining representative values for requirements for each cluster of stakeholders

Once the clusters are formed, the *stakeholders' preferences analysis* technique computes the median preference values that will be used to represent these clusters in the decision. Our technique uses the median value for this purpose, but the decision-maker may deem the mean to be a better representative value for the clusters depending on his judgement and the context in which the decision is being made. We term this representative preference value as the *cluster preference*.

When there are missing values for individual preferences within the stakeholder clusters for a given requirement, our technique computes the median using only available preferences for that requirement to use as cluster preference. However, if no preference has been elicited for a given requirement for a cluster – that is, there is only missing data for the requirement – the cluster preference is left as empty for that cluster and the decision-maker is free to set the value he wants for this requirement for later analysis.

Assigning weights to clusters of stakeholders

Some requirements prioritisation techniques require weights to be assigned to each stakeholders' groupings. In our approach, the assignment of these weights to the *stakeholder clusters* is left to the decision maker. They could, for example, assign these weights through pair-wise comparison using the analytical hierarchical process (AHP) (Saaty 1980).

4.3 Visualizing Clusters of Stakeholders

The *stakeholders' preferences analysis* technique builds a cluster summary table as shown in Table 2 to display summary information about the clusters. This includes the cluster size and their cluster preferences for each requirement.

	Cluster	Size	R1	R2	R3	R4	R5
1	1	1	2	1	5		5
2	2	2	3.5		2	2.5	3
3	3	2	3.5	4.5	4.5	4	-1
4	4	4	1	1	1.5	1	4.5

Table 2 Cluster Summary for running example

In the resulting clusters for the running example, the largest cluster is cluster 4 with 4 stakeholders in it. There is also a single-stakeholder cluster C1. No representative value has been found for R2 in cluster 2 and R4 in cluster 1. Both stakeholders with negative preference have been placed in cluster 3.

The decision maker may also want to check which stakeholder is in which cluster and what their ratings are to better understand the clusters. The *stakeholders' preferences analysis* technique further generates a detailed cluster composition table that lists all the stakeholders together with the cluster they belong to and their rating for the requirements. The detailed cluster composition for the running example is shown in Table 3. The technique has used the value of 1 to represent the preference of Stakeholder I for requirement R2 after the clusters have been generated. This value is been obtained by computing the median of available values for that requirement for that cluster.

	Cluster	R1	R2	R3	R4	R5	Stakeholder
1	1	2	1	5		5	H
2	2	3		1	3	3	F
3	2	4		3	2	3	G
4	3	3	4	5	3	-1	B
5	3	4	5	4	5	-1	D
6	4	1	2	1	2	5	A
7	4	1	1	1	1	5	C
8	4	1	1	2	1	3	E
9	4	1		2	1	4	I

Table 3 Detailed Cluster Composition for running example

4.4 Impact on Decision Making

To assess the benefits of clustering stakeholders according to their preferences, we evaluate how clustering affects decision making compared to alternative approaches which consist in either viewing all stakeholders as forming a single large homogeneous group or grouping stakeholders according to characteristics such as their role in the organisation. For each of these three different approaches, we compare the stakeholders' individual preferences and the preferences of the group they belong to.

Clustering the stakeholders according to their preferences means that the divergence between cluster preferences and the individual preferences of the stakeholders should be smaller than with the other approaches. We argue that this leads to better decision as the

preferences used to represent the stakeholders in the decision is closer to their actual individual preferences.

We also look at how trends and outliers can be identified from our clusters of stakeholders. We build pie charts to show how the composition of clusters varies and box plots to show how the ratings of the stakeholders vary within and among the clusters. Outliers are single stakeholder clusters who have been left of their own because their ratings are too different from the clusters.

4.4.1 Impact of Cluster Preference on Requirements Decision Input

We proceed by measuring how close the cluster preference values used to represent the individual stakeholders are to their actual preferences compared to the preferences that would have been used if there was no clustering.

As defined in section 4.2.3, if there are m stakeholders rating n requirements, the individual preferences of the stakeholders is represented by a $m \times n$ matrix P as follows:

$$\mathrm{P} = \begin{bmatrix} p_{11} & \cdots & p_{1j} \\ \vdots & \ddots & \vdots \\ p_{i1} & \cdots & p_{mn} \end{bmatrix}$$

where p_{ij} is the *individual preference* of stakeholder i for the requirement j.

If k *stakeholder clusters* are discovered during the *stakeholders' preferences analysis* then the preference matrix is defined as follows:

$$P_c = \begin{bmatrix} p_{c11} & \cdots & p_{c1n} \\ \vdots & \ddots & \vdots \\ p_{ck1} & \cdots & p_{ckn} \end{bmatrix}$$

where p_{cij} is the *cluster preference* of stakeholder cluster i for the requirement j.

If there are l stakeholder groups, then the preference matrix for these groups is given by:

$$P_g = \begin{bmatrix} p_{g11} & \cdots & p_{g1n} \\ \vdots & \ddots & \vdots \\ p_{gl1} & \cdots & p_{gln} \end{bmatrix}$$

where p_{gij} is the *group preference* for stakeholder group i for the requirement j.

The overall preference matrix for all stakeholders is given by:

$$P_o = [p_{o1} \quad \cdots \quad p_{on}]$$

Where p_{oj} is the *overall preference* of all stakeholders for the requirement *j*. In this case we compute these values as the median of all preferences for all stakeholders.

We first proceed by computing the following Gower's distances for each stakeholder *i*:

- $divergence_o$, the divergence between overall preference and the individual preference of stakeholder i.

$$divergence_o = D_G([p_{i1}, \dots, p_{in}], [p_{o1}, \dots, p_{on}])$$

- $divergence_g$, the divergence between group preference of the stakeholder group to which stakeholder i belongs and the individual preference of stakeholder i

$$divergence_g = D_G([p_{i1}, \dots, p_{in}], [p_{gx1}, \dots, p_{gxn}])$$

where x is the stakeholder group to which stakeholder i belongs.

- $divergence_c$, the divergence between cluster preference of the stakeholder cluster to which stakeholder i belongs and the individual preference of stakeholder i

$$divergence_c = D_G([p_{i1}, \dots, p_{in}], [p_{cy1}, \dots, p_{cyn}])$$

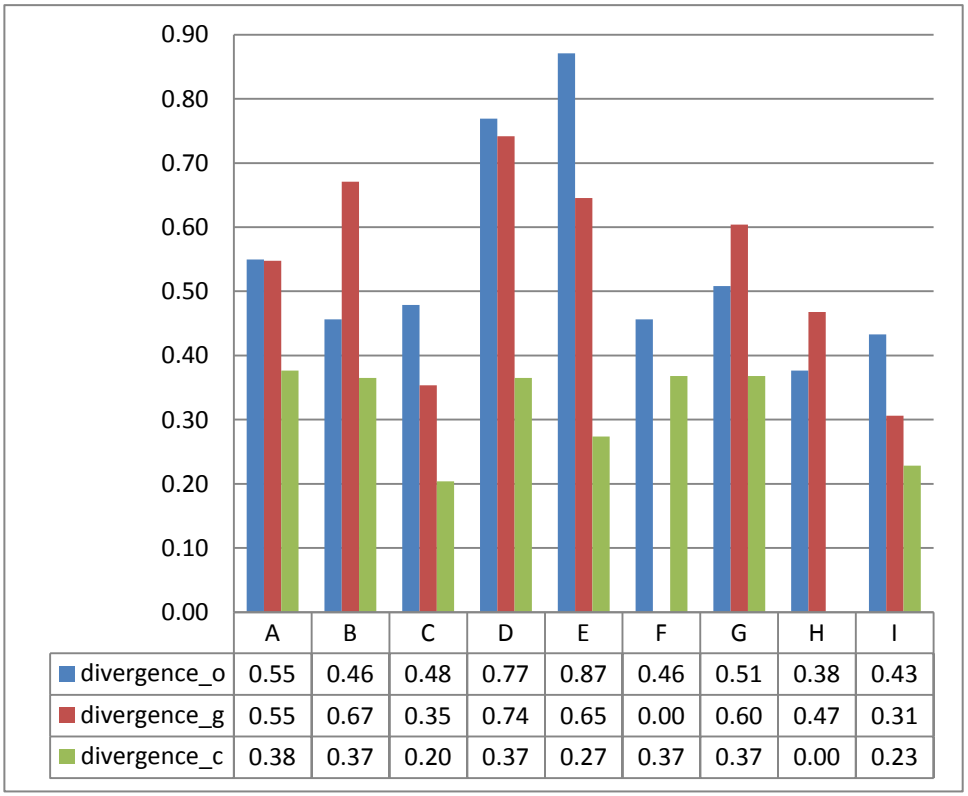
where y is the stakeholder cluster to which stakeholder i belongs.

The resulting Gower's distances for the running example are listed in Figure 18. We can see that $divergence_c$ tends to be lower than both $divergence_g$ and $divergence_o$ for the stakeholders, indicating that the cluster preferences are closer to the individual preferences. The only exception is Stakeholder F for whom $divergence_c$ is higher than $divergence_g$ but nonetheless remains less than $divergence_o$. Another observation is that $divergence_c$ for Stakeholder H is 0 since Stakeholder H is a cluster on his own.

We build the box plots of the distances $divergence_o$ $divergence_g$ and $divergence_c$ (as shown in Figure 19) to get an overall idea of how they compare. Box plots are used to

display sets of data in a single graph (Tukey 1977). The box plot uses five values from the dataset- these are the extremes, the upper and lower quartiles and the median. The width of the box shows where the middle 50% of the data (between the lower quartile and the lower quartile) lies. A large box means that the middle 50% of the data is widely spread while a small box means middle 50% of the data is within a smaller range. The whiskers of the box plot show where the extremes of the data lie. The position of the box between the whiskers shows the skewness of the dataset.

In the running example, median of $divergence_c$ is less than that of $divergence_o$ and $divergence_g$. The spread of $divergence_c$ is also less than that of both $divergence_o$ and $divergence_g$. This shows that the cluster preferences are closest to the actual preference of the stakeholders and hence, by using this value in the decision we better represent the preference of the stakeholder in it.



	A	B	C	D	E	F	G	H	I
divergence_o	0.55	0.46	0.48	0.77	0.87	0.46	0.51	0.38	0.43
divergence_g	0.55	0.67	0.35	0.74	0.65	0.00	0.60	0.47	0.31
divergence_c	0.38	0.37	0.20	0.37	0.27	0.37	0.37	0.00	0.23

Figure 18 Gower's distances between individual preference and overall, stakehololder group and cluster preferences

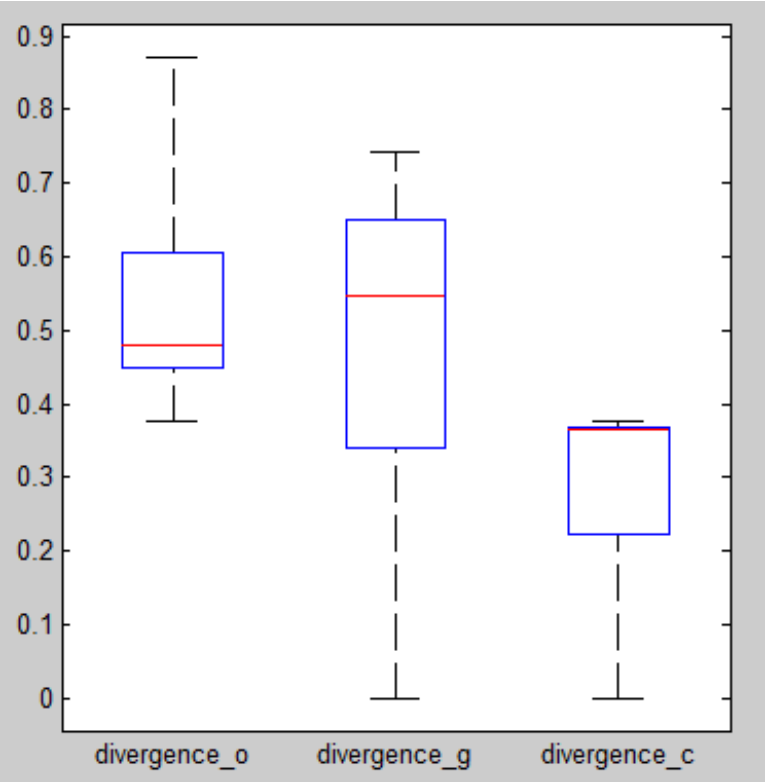


Figure 19 Box plot of preference divergences

4.4.2 Trends Analysis

One of the benefits of our technique is that the groups of stakeholders that have been identified can be further analyzed to identify trends and diverging opinions. It is useful to understand any information that the clusters may uncover about the stakeholders based on their preferences. The output from the *stakeholders' preferences analysis* can be used to understand the clusters better using simple techniques. For example, for each cluster, we can visualise the spread of the stakeholders' preferences for each requirement using box plots of the requirements preferences for all the stakeholders in that cluster.

Figure 21 shows such a visualisation for our running example where we have more than one stakeholder. The plots are ordered in order of cluster identifiers; cluster 2, cluster 3 and cluster 4. On the box plots, the requirements are sorted in order of cost, with the cheapest one first and more expensive one last. When there is only on stakeholder in the cluster (as it is the case for cluster 2), we can simply plot the preference for each requirement as shown in Figure 22.

This gives the decision maker interesting insights. For example, for the largest cluster 4, the stakeholders' preferences have small boxes for R1, R3, R2 and R4, but the box for

R5 is larger, ranging between 3 and 5. This means that the middle 50% of stakeholders in cluster 3 tend to agree on R1, R3, R2 and R4 while they have more diverging preferences for R5. Comparing all the clusters, it can be observed that cluster 2 has the highest preference for requirement R1 and cluster 4 has the lowest preference for requirement R5. Comparing the clusters' box plots to the box plots of the preferences of all the stakeholders in Figure 20, it can be seen that the preference spread decreases for the requirements when clustering the stakeholders by preference.

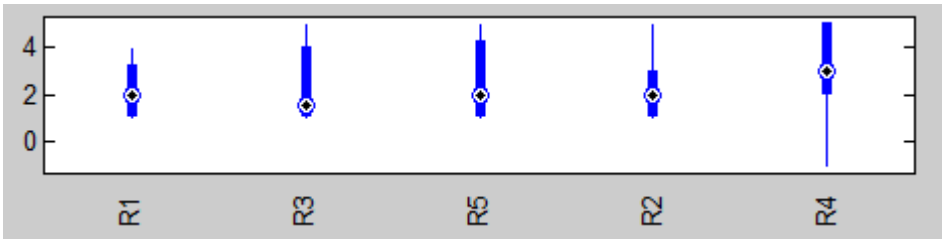


Figure 20 Box plot for distribution of preferences for all stakeholders

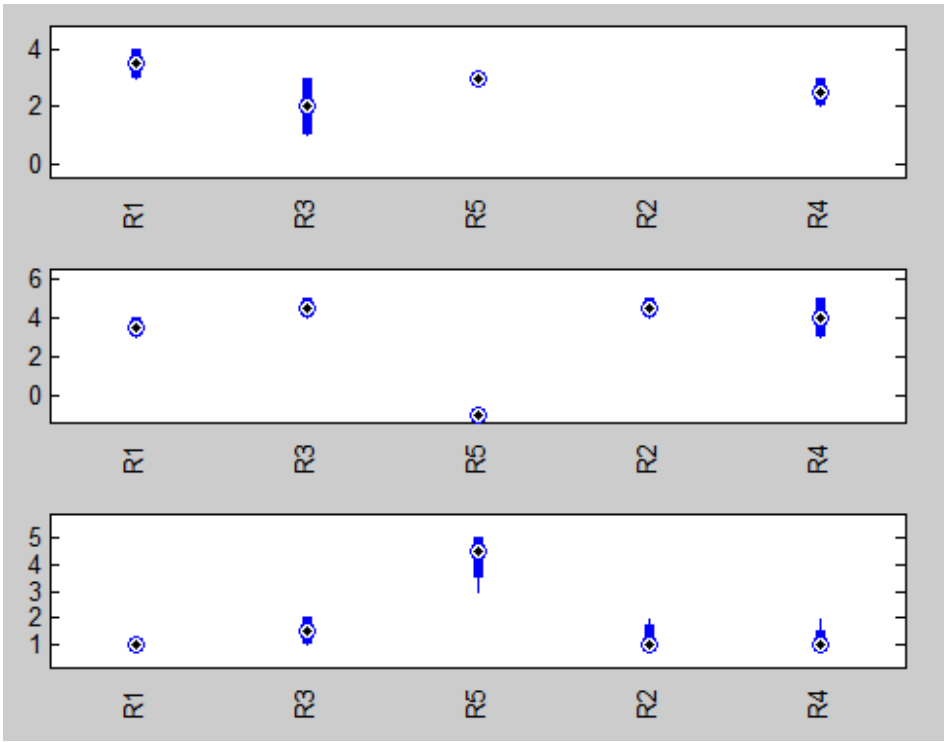


Figure 21 Box plot for distribution of preferences in clusters cluster 2, cluster 3 and cluster 4

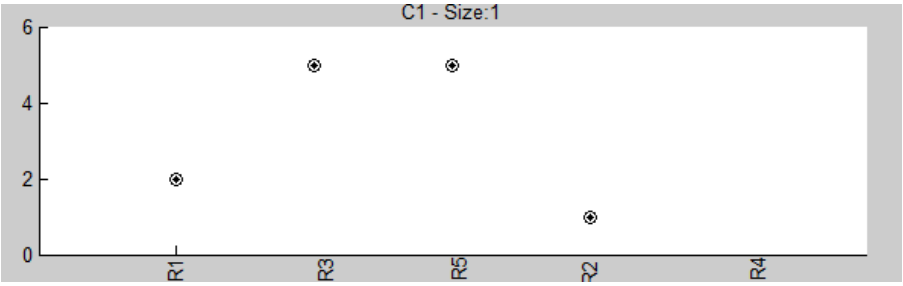


Figure 22 Box plot for distribution of preferences in cluster 1

Where more extensive information about the stakeholders is available (for example, their age and professional position) the decision maker can further analyse the clusters using pie charts to understand how the clusters are composed and identify any exception. In the running example, the information that we have about the stakeholder groups (G1-G3) can be used for this analysis. The pie chart for the largest stakeholder cluster 4 (Figure 23) shows that it has stakeholders from all the stakeholder groups in it. Similarly the pie chart for the largest stakeholder group G2 (Figure 24) shows that these stakeholders are not only in cluster 4 but they are also in clusters 1 and 2 as well. The decision maker may want to further investigate why the preferences of these stakeholders who are in a single stakeholder group are so dissimilar.

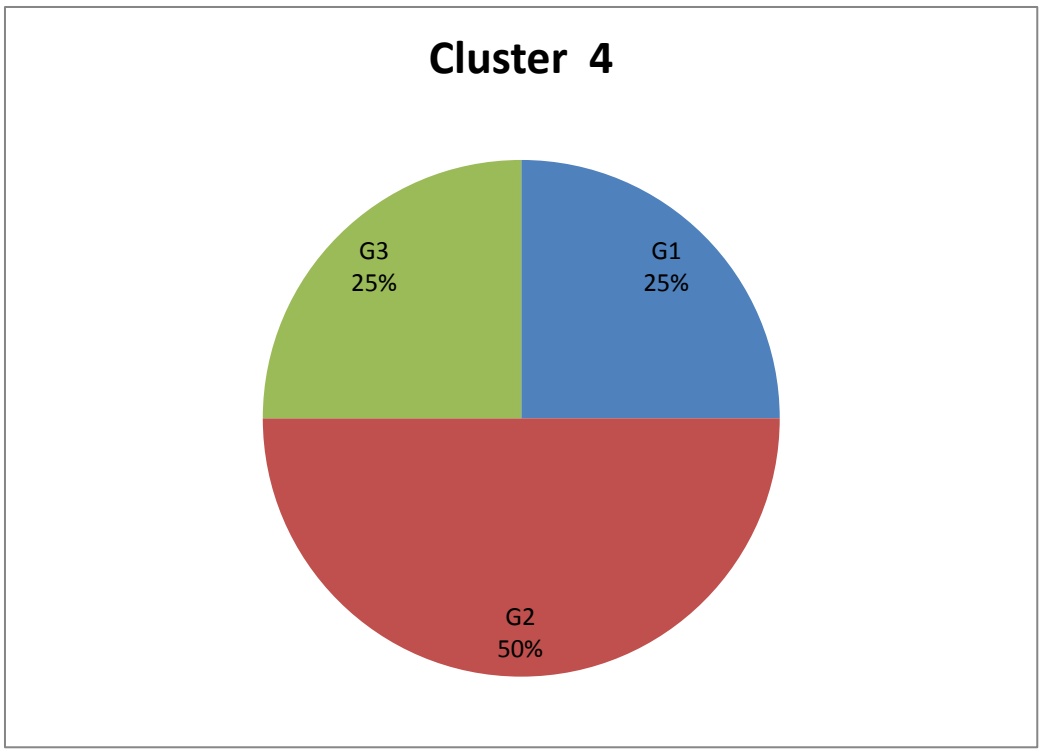


Figure 23 Pie chart showing how stakeholder groups are distributed in Cluster 4

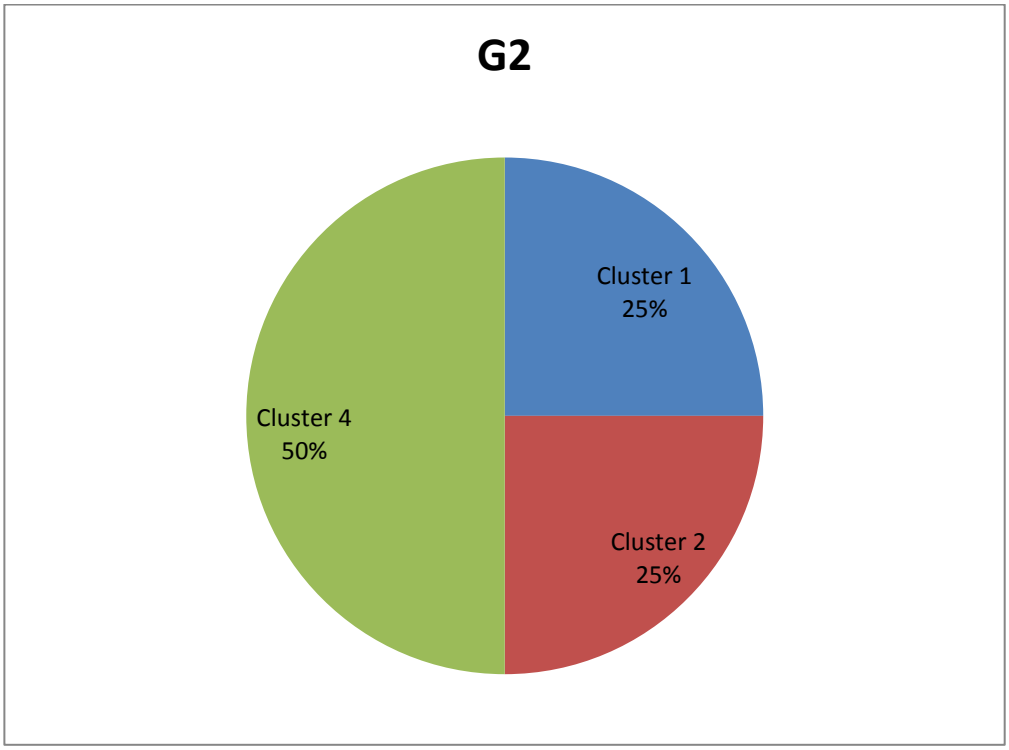


Figure 24 Pie chart showing how stakeholders in G2 are distributed among the clusters

4.4.3 Outlier Analysis

One question that arises when there outlying stakeholders is whether or not to include them in the decision process. Clusters with one single stakeholder in them are identified as outliers. This does not mean that these 'outlier' stakeholders need to be ignored. On the contrary, they may bring new insights on why they have such diverging preferences. In the running example, one such outlier is Stakeholder H. He is the only stakeholder who has high ratings for both R3 and R5.

When such outlying stakeholders are present in the clusters, the decision maker may decide to further investigate why this is the case by contacting the stakeholders and investigate why they have rated the requirements the way they have. This may enable the decision maker to find new assumptions or constraints or even elicit further requirements that he may have overlooked.

If the stakeholders cannot be contacted for further investigation, the decision maker may choose to include or exclude some or all of these outlying stakeholders before using the actual decision making technique. This exercise is again a subjective one and will depend on the data that is available to the decision maker. He may for example determine

the importance of a stakeholder by computing their pagerank (Page et al. 1998; Brin and Page 1998; Lim 2010) or power within a project (Milne and Maiden 2011) and decide whether to include or exclude him based on the resulting value. This kind of analysis is feasible when tools like Stakesource (Lim 2010) are used as they also store the required information to compute importance of stakeholders. If such information is not available, the decision maker may use his own judgment to give an importance to the outlying stakeholders.

4.4.4 Impact of Cluster Preferences on Requirements Decisions Results

The *stakeholders' preferences analysis* generally produces *cluster preferences* that are closer to *individual preferences*. This means that if we use stakeholder clusters and their corresponding *cluster preferences* as input in requirements prioritisation techniques we get a better representation of the preference of the stakeholders in them. We argue that this leads to better, or at least better informed decisions than the alternative approaches. However, we have no objective means to assess the impact of using *cluster preferences* on the results of requirements decisions as the decisions can be different depending on other factors that the decision makers have taken into consideration. For example, the weights given to the different clusters of stakeholders can considerably influence the results.

4.5 Other Uses of Stakeholder Preferences Analysis

The *stakeholders' preferences analysis* technique can be particularly useful in the field of User-Centered System Design (Noyes and Baber 1999) where the focus is on the needs of the users of the system. The usual first step in this process is the discovery of the target user segments of the systems being designed (Kramer, Noronha, and Vergo 2000). This can be performed using market segmentation techniques but since our technique is similar to those techniques but geared towards requirements, there is a potential application of it in this field.

Another field of interest could be that of context-aware systems (Schilit, Adams, and Want 1994). Since our technique allows us to do further analysis on the attributes of stakeholders such as location, age etc. (if the data is available before hand), this can be used to find one or more clusters of stakeholders that relate to a specific context and hence know what users expect in that specific context based on the representative values for the requirements for the clusters.

4.6 Tool Support for Clustering Stakeholders

We have developed a tool in Matlab to enable decision makers to perform *stakeholders' preferences analysis* easily. It takes as input the stakeholder ratings in an Excel spread sheet and performs cluster analysis on the dataset after choosing the best inter-cluster similarity measure. The output is the dendrogram, the resulting clusters along with their corresponding cluster preferences and graphs to enable further analysis of the clusters. The user can select how the final clusters are formed by either using the default cut-off value or manually choose one cut-off value that makes more sense in the context of the analysis.

This tool is further described in Appendix A.

4.7 Related Work

Several works in the field of requirements engineering have used clustering algorithms to try to identify groups in the large volumes of data that are collected during the elicitation stage. For example, the tool Pirogov uses hierarchical clustering to group requirements into categories based on the terms in them (Laurent, Cleland-Huang, and Duan 2007). It places requirements into multiple orthogonal categories that take into account the roles played by individual requirements. One clustering technique organizes requirements by feature sets while others cluster requirements around user-defined themes such as business goals and high level use cases. Every requirement is thus placed into one or more feature set. Cross-cutting subsets of requirements are placed into additional categories. Stakeholders determine the relative value of each cluster and weigh the importance the clustering methods. The tool then uses objective functions to generate the final prioritisation decisions.

'Organizer & Promoter of Collaborative Ideas" – OPCI (Castro-Herrera, Cleland-Huang, and Mobasher 2009), also uses clustering to find groups of stakeholders with similar interests on from their discussions on forums. It first clusters stakeholders based on the words in their posts and build stakeholder profiles for the stakeholders. OPCI then recommends forums that may be of interest to the stakeholders based on their profiles. This practice encourages more constructive and efficient contribution of the stakeholders during the requirements elicitation phase as they are more likely to discuss about the features or requirements that are relevant to them.

Another example of application of clustering in the field of requirements engineering is the work of Duan et al in the Poirot framework where they have investigated the use of bisecting divisive clustering algorithms to support traceability in requirements engineering (Duan and Cleland-Huang 2007). The technique uses clustering to group artefacts such as requirements and java classes in the system by similarity. When the user makes a trace query, the technique will compute probability scores and group candidate links according to previously generated clusters. Each cluster is given a meaningful name and ranked according to their relevance to the query. The technique also attempts to include recessive clusters in the results which are then displayed to the user.

None of these works addresses the problem of clustering stakeholders based on their preferences in order to facilitate the requirements selection and prioritisation as we have done in this thesis.

4.8 Conclusion

The *stakeholders' preferences analysis* is particularly useful in the context where we are eliciting requirements and preferences for these requirements from a large number of stakeholders. It takes as input individual stakeholders' values for a set of requirements to be evaluated, and generates as output a set of stakeholder clusters together with their corresponding cluster preferences for each requirement. These cluster preferences can then be used by existing decision-making techniques to rank the requirements or generate a Pareto front.

Grouping stakeholders with similar ratings together results in cluster preference values for each requirement that are close to the ratings of its cluster members. When we use these cluster preferences as input to requirements decision-making techniques, we make better decisions that better reflect the needs of the stakeholders. Furthermore, these clusters of stakeholders can be further analysed to understand why the stakeholders have rated requirements the way they have and give the decision-maker further insights about the stakeholders and the requirements themselves.

We have not investigated how semantic relationships among requirements are reflected in our stakeholders' preferences analysis technique. It will be interesting to see how the preferences of stakeholders vary according to dependencies, functional similarities or conflicts among the requirements by clustering the requirements according to the ratings provided by stakeholders instead.

Chapter 5 - Optimal Solutions Analysis

We apply clustering algorithms to identify similarities among solutions on the Pareto Front.

5.1 Introduction

Some requirements decision making techniques generate as output Pareto Optimal fronts that depict the sets of optimal solutions. The complexity of the Pareto optimal front increases when we have a large number of requirements making it very difficult to understand the similarities and differences across the solutions in these cases.

In this context, a solution is a set of requirements to be included in the next release of the software system being investigated. In the most common scenario, the decision maker will have two objectives: reduce cost and maximize the value of the requirement to the stakeholders. To be able to make the right choice of solution in this case, decision makers need to know the following information:

1) In a given cost or value range, how do the optimal solutions differ and how are they similar?
2) Are there significantly different solutions that are close to each other on the Pareto optimal front?

The *optimal solutions analysis,* depicted in Figure 25, enables the decision makers to make better informed decisions about which solution to implement as it facilitates the analysis of the design options in the different solutions on the Pareto front. It uses clustering algorithms to group solutions with *similar design,* i.e. selected requirements, on the Pareto Optimal front. Our work looks at solution sets that are represented as bit vectors where a 1 indicates presence of a requirement and 0 represents the absence of a requirement.

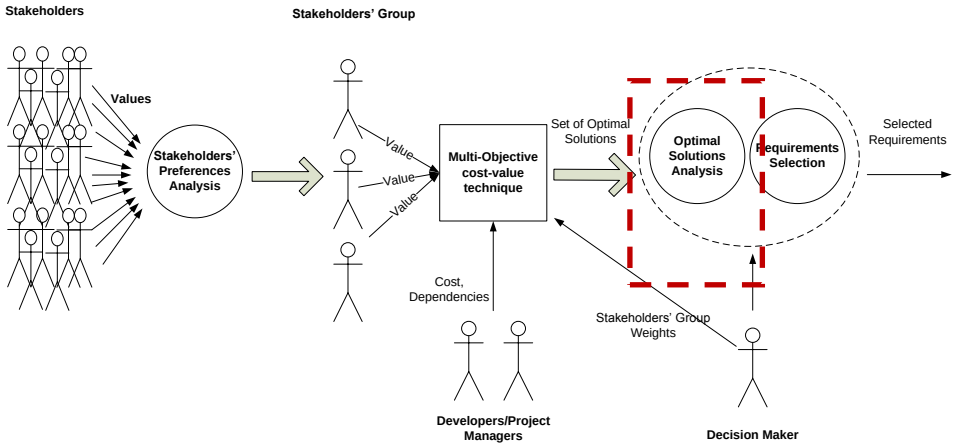


Figure 25 Context of optimal solutions analysis

Our objective is different from the objectives of analysing the sensitivity of optimal solutions with respect to variations in the model parameters – e.g. the cost and value of individual requirements (Harman et al. 2009) – or analysing the robustness of the solutions with respect to the non-determinism of the genetic algorithm search (Gay et al. 2010). Sensitivity and robustness analysis are concerned with understanding how the set of optimal solutions could change due to uncertainties in the model parameters or caused by the genetic algorithm; we are concerned with helping decision makers understand the set of generated solutions. Our objective here is therefore orthogonal and complementary to sensitivity and robustness analysis.

We have implemented a tool in Matlab that takes as input the set optimal solutions and produces the clusters of similar solutions and related graphical displays for further analysis.

5.2 Grouping Solutions by Design Similarity

In Chapter 2, we have seen that although there already are post-Pareto analysis techniques described in literature, none have looked at how to interpret the design variations on the Pareto optimal front. All of these focus on how the solutions attain the objectives being investigated. But in the field of requirements engineering, enhancing the analysis of objective attainment with possible design variations is more useful. This gives the decision maker greater insights over how he may attain similar objectives with different designs.

The *optimal solutions analysis* technique does not take into account how the Pareto Optimal solutions have been found. The formulae for computing the objective functions

i.e. total cost and values, from estimated parameters can vary. This is the case for example for the DDP framework (Feather and Menzies 2002). Our work is relevant wherever techniques produce a potentially large set of optimal or near optimal solutions to a multi-objective problem.

Running example

To illustrate how we use clustering on solutions we use the following running example with 8 requirements whose cost and values are given in Table 4. Applying the NSGA-II optimization technique described in (Zhang 2010) generates a set of 27 solutions which are shown in Table 5. Figure 26 plots all 27 Pareto optimal solutions in terms of their cost and value.

	R1	**R2**	**R3**	**R4**	**R5**	**R6**	**R7**	**R8**
Value	5	3	3	2	2	1	1	1
Cost	10	5	6	5	6	3	2	3

Table 4 Cost and Value Table

Solution	**R1**	**R2**	**R3**	**R4**	**R5**	**R6**	**R7**	**R8**	**Value**	**Cost**
S1	0	0	0	0	0	0	1	0	1	2
S2	0	1	0	0	0	0	0	0	3	5
S3	0	1	0	0	0	0	1	0	4	7
S4	0	1	0	0	0	1	1	0	5	10
S5	0	1	0	1	0	0	0	0	5	10
S6	1	0	0	0	0	0	0	0	5	10
S7	0	1	0	0	0	0	1	1	5	10
S8	0	1	1	0	0	0	0	0	6	11
S9	0	1	1	0	0	0	1	0	7	13
S10	1	1	0	0	0	0	0	0	8	15
S11	1	1	0	0	0	0	1	0	9	17
S12	1	1	0	0	0	1	1	0	10	20
S13	1	1	0	1	0	0	0	0	10	20
S14	1	1	0	0	0	0	1	1	10	20
S15	1	1	1	0	0	0	0	0	11	21
S16	1	1	1	0	0	0	1	0	12	23
S17	1	1	1	0	0	1	1	0	13	26
S18	1	1	1	0	0	0	1	1	13	26
S19	1	1	1	1	0	0	0	0	13	26
S20	1	1	1	1	0	0	1	0	14	28
S21	1	1	1	1	0	1	1	0	15	31
S22	1	1	1	1	0	0	1	1	15	31

S23	1	1	1	1	1	0	1	0	16	34
S24	1	1	1	1	0	1	1	1	16	34
S25	1	1	1	1	1	0	1	1	17	37
S26	1	1	1	1	1	1	1	0	17	37
S27	1	1	1	1	1	1	1	1	18	40

Table 5 Set of Pareto Optimal Solutions for running example

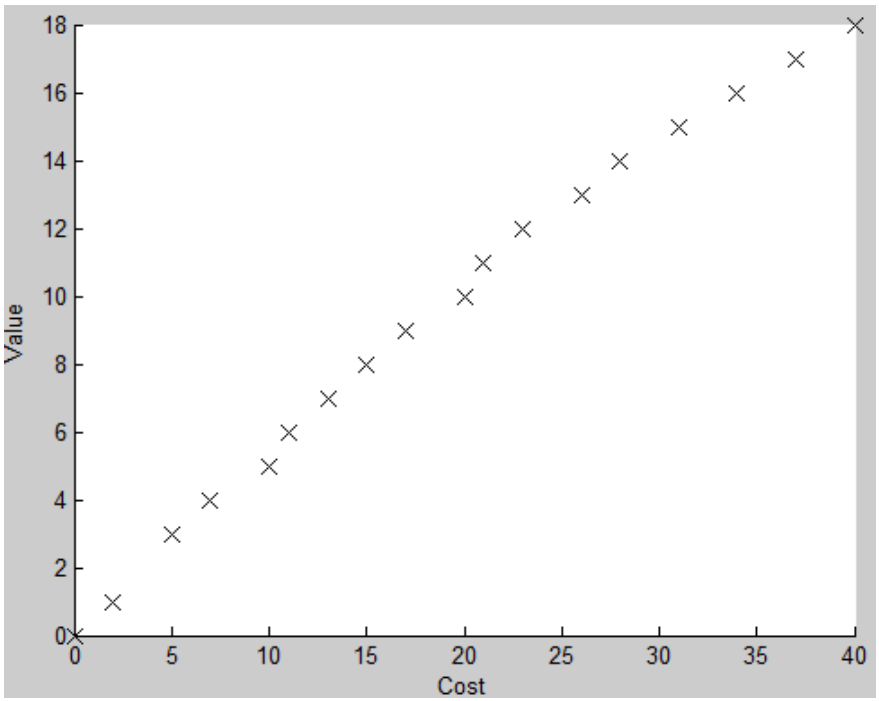


Figure 26 Pareto Optimal Front

5.2.2 Distance Measures for Solutions

We are here comparing solutions in terms of design similarities. Since the solutions are bit vectors, our distance measure must be able to compute the distance between binary data.

Different distance measures can be used to compare bit vectors. A well-known measure is the Hamming distance (Hamming 1950) that counts the number of bits that are different between two vectors. For example, the Hamming distance between S4 and S5 is 3 because they disagree on 3 requirements selection decisions (R4, R6, and R7). Similarly, the Hamming distance between S17 and S19 is also 3. Such measure, however, is not ideal in our context because it gives the same importance to requirements selection and rejection. This has undesirable consequences when assessing how close two solutions are to each other.

To illustrate this, let us consider an example where there are 20 requirements to choose from and two solutions that each includes 2 requirements, one of which is common to both solutions and one that differs as shown in Table 6 below:

R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17	R18	R19	R20
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 6 Example with 20 requirements - 3 requirements included

The Hamming distance between these solutions is 2. If the same third requirement is added to both solutions (so that they both include 2 requirements in common and differ on the two other) as in Table 7, their Hamming distance is still 2 despite the fact that they now have one more requirement in common.

R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17	R18	R19	R20
1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 7 Example with 20 requirements - 4 requirements included

If we keep on adding common requirements to both solutions so that they each include the same 18 requirements, for example, and differ on their selection of the 2nd and 3rd requirements only (shown in Table 8), their Hamming distance will still be 2 despite the fact that they are now very similar in terms of the development activities they would require and only differ on a small fraction of what would need to be done.

R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15	R16	R17	R18	R19	R20
1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Table 8 Example with 20 requirements - all requirements included

Jaccard Distance

A distance measure more suitable for the *optimal solutions analysis* technique is the Jaccard distance (Jaccard 1908). This distance measures the overlap in the number of selected requirements between two solutions. For example, if there are two solutions x and y both with n requirements, the Jaccard distance between them is:

$$Dist(x, y) = 1 - \frac{\sum_{i=1\ldots n} x_i \wedge y_i}{\sum_{i=1\ldots n} x_i \vee y_i}$$

More simply, in set-theoretic terms, if x and y denote sets of selected requirements, their Jaccard distance is given by 1 minus the ratio of the number of requirements they have in common over the number of all requirements present in either of the solutions:

$$Dist(x, y) = 1 - \frac{\#\, x \cap y}{\#\, x \cup y}$$

The Jaccard distance between two solutions is always a number between 0 and 1. Thus, the Jaccard distance between solutions S4 and S5 is given as follows:

$$Dist(S4, S5) = 1 - \frac{\sum_{i=1\ldots 8} S4_i \wedge S5_i}{\sum_{i=1\ldots 8} S4_i \vee S5_i}$$

$$Dist(S4, S5) = 1 - \frac{1}{4} = \frac{3}{4}$$

Where the Jaccard distance between solutions S17 and S19 is computed as follows:

$$Dist(S7, S9) = 1 - \frac{\sum_{i=1\ldots 8} S7_i \wedge S9_i}{\sum_{i=1\ldots 8} S7_i \vee S9_i}$$

$$Dist(S17, S19) = 1 - \frac{3}{6} = \frac{1}{2}$$

The Jaccard distance has found S17 to be closer to S19 than S4 is to S5 as S17 and S19 have more common selected requirements than S4 and S5.

Considering the example where there are 20 requirements to choose from and two solutions that each includes 2 requirements (as shown in Table 6), the Jaccard distance between these two solutions is:

$$Dist = 1 - \frac{1}{3} = \frac{2}{3}$$

If the same third requirement is added to them as in Table 7, their Jaccard distance becomes:

$$Dist = 1 - \frac{2}{4} = \frac{1}{2}$$

The Jaccard distance is now less than before as the Jaccard distance accounts for the fact that the two solutions now have one more requirement in common.

Thus, if we keep adding common requirements to both solutions until they each include the same 18 requirements, on their selection of the 2nd and 3rd requirement only (shown in Table 8), their Jaccard distance will be:

$$Dist = 1 - \frac{18}{20} = \frac{1}{10}$$

This distance reflects the fact that these solutions are now very similar in terms of the development activities they would require and only differ on a small fraction of what would need to be done.

Weighted Jaccard Distance

The Jaccard distance gives the same importance to all requirements when comparing solutions. In practice, this is generally not the case; some requirements will have much more impact on the system design than others. For example, if we are making design decisions for a mobile phone, a requirement to include a touch screen has much more impact on the system design than a requirement for the phone to have an alarm. This is because the complexity of implementation of each requirement varies greatly. When assessing how close solutions are to each other, two solutions that include both the touch screen requirement and differ only on their inclusion or not of the alarm should be seen as closer to each other than two solutions that both include the alarm but differ on their inclusion of a touch screen. To allow for the importance of requirements to be taken into account, we will use a weighted version of the Jaccard distance:

$$Dist(x, y) = 1 - \frac{\sum_{i=1\ldots n} w_i x_i \wedge w_i y_i}{\sum_{i=1\ldots n} w_i x_i \vee w_i y_i}$$

Where $[w_1\ w_2\ \ldots\ w_n]$ is a weight vector where w_i represents the weight of requirements r_i in the design decision.

The decision makers can choose any weight vector to reflect the importance they want to give to requirements when assessing the distance between two solutions. One

approach can be to give high weights to requirements that have a high impact on the system architecture and low weight to those that have low impact on the architecture. Another one that does not require additional input from decision makers is to use the cost vector as a measure of the importance of a requirement on the system design. The clustering algorithm will therefore tend to group together solutions that have a lot of development effort (or cost) in common. This reflects our perspective that the decision makers are the product developers. If the decision making process is viewed from the perspective of a product user, a good choice for the weight vector could be to use the requirements values.

For example, if we look at solutions S1, S2 and S3 from our running example, if we use the cost as the weight vector, we compute the weighted Jaccard distance as follows:

$$Dist(S1, S3) = 1 - \frac{2 \times 1}{5 \times 1 + 2 \times 1} = \frac{5}{7}$$

$$Dist(S2, S3) = 1 - \frac{5 \times 1}{5 \times 1 + 2 \times 1} = \frac{2}{7}$$

As both S3 and S2 include R2 which has higher cost than R7, they are closer to each other and hence more “similar”.

5.2.3 Inter-cluster Similarity Measure

The candidate inter-cluster similarity measures for the *optimal solutions analysis* technique are the average, weighted-average linkage and the complete linkage algorithms. These three have the most consistent performance and are more likely to produce more reliable clusterings.

The *optimal solutions analysis* technique applies all the candidate inter-cluster similarity measures and then computes the Cophenetic Correlation coefficient in each case to determine which measure gives the higher value for the coefficient. It then uses the clusterings with the higher Cophenetic Correlation coefficient for the final clusters.

5.2.4 Generating Clusters of Optimal Solutions

The *optimal solutions analysis* technique uses the hierarchical agglomerative algorithm to cluster solutions according to their design similarities. The hierarchical agglomerative algorithm starts with all solutions on the Pareto front as individual

clusters. Then it merges solutions with similar designs into clusters in a sequential way until all the solutions are in a single cluster.

For the running example, the *optimal solutions analysis* technique determines the inter-cluster distance measure to use from the Cophenetic Correlation coefficient for each of the average, weighted average and complete linkages. These are as follows:

$Ccc_{Average\ Linkage}$: 0.9228

$Ccc_{Weighted\ Average\ Linkage}$: 0.9104

$Ccc_{Complete\ Linkage}$: 0.8730

A larger Cophenetic Correlation coefficient for the average linkage means that it will produce the more realistic clusters in this case. Figure 27 illustrates the dendrogram resulting from the use of the average linkage. The x-axis labels are numbers representing the solutions in the order in which they are in the dataset fed to the clustering algorithm.

5.2.5 Identifying the Clusters of Solutions

The Mojena cut-off is then used to propose a default cut-off on the dendrogram to form the clusters. In the case of our running example, the default cut-off suggests 6 clusters. If the decision maker wishes to check the quality of the clusters, he can refer to the Rand, C and Silhouette indices for the clusters. As shown in Figure 28, the default number of clusters is a reasonable choice but the cluster quality can be further improved by increasing the number of clusters from 6 to 7.

Table 9 shows the detailed composition of the 7 clusters. However, since the *optimal solutions analysis* technique uses hierarchical clustering algorithm, the decision maker can choose to cut-off the dendrogram at any distance he feels fit his purpose and further verify the cluster quality until he gets good clusters that satisfies him.

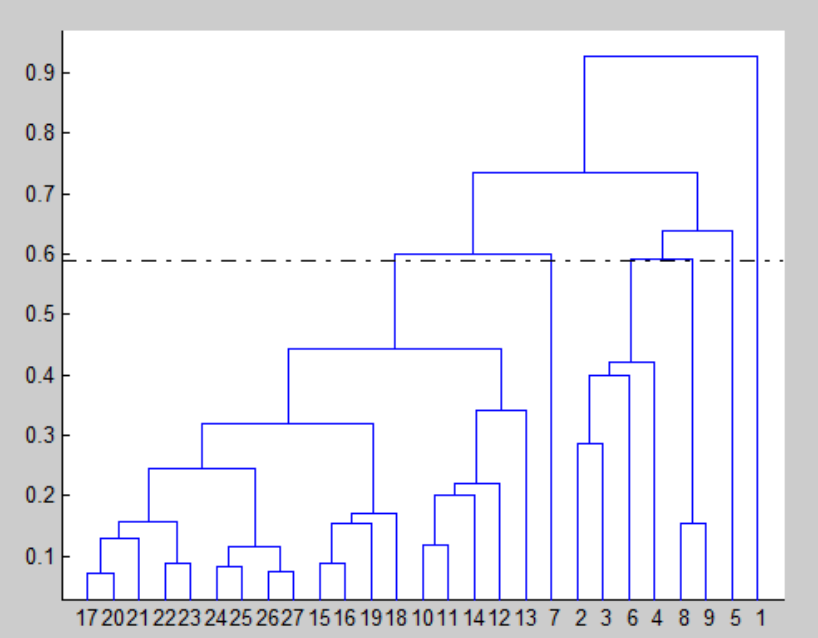


Figure 27 Dendrogram for running example - weighed by cost

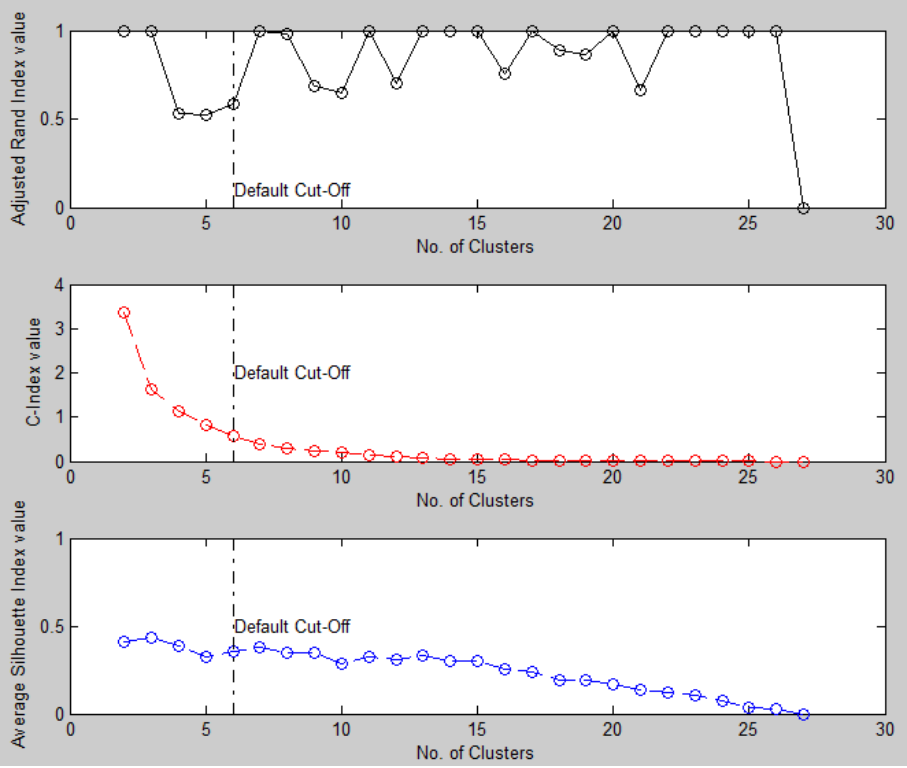


Figure 28 Quality Indices for clusters for running example

Cluster	Solution	R1	R2	R3	R4	R5	R6	R7	R8	Value	Cost
C1	**S15**	1	1	1	0	0	0	0	0	11	21
C1	**S16**	1	1	1	0	0	0	1	0	12	23
C1	**S17**	1	1	1	1	0	0	0	0	13	26
C1	**S18**	1	1	1	0	0	0	1	1	13	26
C1	**S19**	1	1	1	0	0	1	1	0	13	26
C1	**S20**	1	1	1	1	0	0	1	0	14	28
C1	**S21**	1	1	1	1	0	0	1	1	15	31
C1	**S22**	1	1	1	1	0	1	1	0	15	31
C1	**S23**	1	1	1	1	0	1	1	1	16	34
C1	**S24**	1	1	1	1	1	0	1	0	16	34
C1	**S25**	1	1	1	1	1	1	1	0	17	37
C1	**S26**	1	1	1	1	1	0	1	1	17	37
C1	**S27**	1	1	1	1	1	1	1	1	18	40
C2	**S8**	0	1	1	0	0	0	0	0	6	11
C2	**S9**	0	1	1	0	0	0	1	0	7	13
C3	**S7**	1	0	0	0	0	0	0	0	5	10
C4	**S5**	0	1	0	1	0	0	0	0	5	10
C5	**S1**	0	0	0	0	0	0	1	0	1	2
C6	**S10**	1	1	0	0	0	0	0	0	8	15
C6	**S11**	1	1	0	0	0	0	1	0	9	17
C6	**S12**	1	1	0	0	0	0	1	1	10	20
C6	**S13**	1	1	0	1	0	0	0	0	10	20
C6	**S14**	1	1	0	0	0	1	1	0	10	20
C7	**S2**	0	1	0	0	0	0	0	0	3	5
C7	**S3**	0	1	0	0	0	0	1	0	4	7
C7	**S4**	0	1	0	0	0	0	1	1	5	10
C7	**S6**	0	1	0	0	0	1	1	0	5	10

Table 9 Detailed Cluster Composition

5.3 Analysing Clusters of Optimal Solutions

Decision makers can use visualisations of the generated clusters to gain insights about the set of optimal solutions. We have developed a set of views intended to help decision makers identifying useful information about the optimal solutions.

5.3.1 Clusters' Distribution on the Pareto front

The first view consists in visualizing the clusters on the Pareto front as shown in Figure 29. Each cluster is delimited by an ellipse of a different colour whose major diameter joins the lowest and highest cost solutions in the cluster and the minor diameter is proportional to the value range of the cluster. Each solution point belonging

to a cluster is also coloured in its cluster's colour. In Figure 29, some solutions points, such as the 4 solutions of cost 10 and value 5, are superimposed on the Pareto Front.

This may also happen for clusters. For example, clusters C3 and C4, two singleton clusters composed of solutions S7 and S5, respectively, are superimposed. An alternate visualization of this graph allows us to separate these superimposed clusters. In this visualization, the y-axis represents the distance of the clusters from C1 and the x-axis is the cost. It enables the decision maker to see how the clusters are superimposed on each other. Figure 30 shows the alternative view of the clusters for our running example.

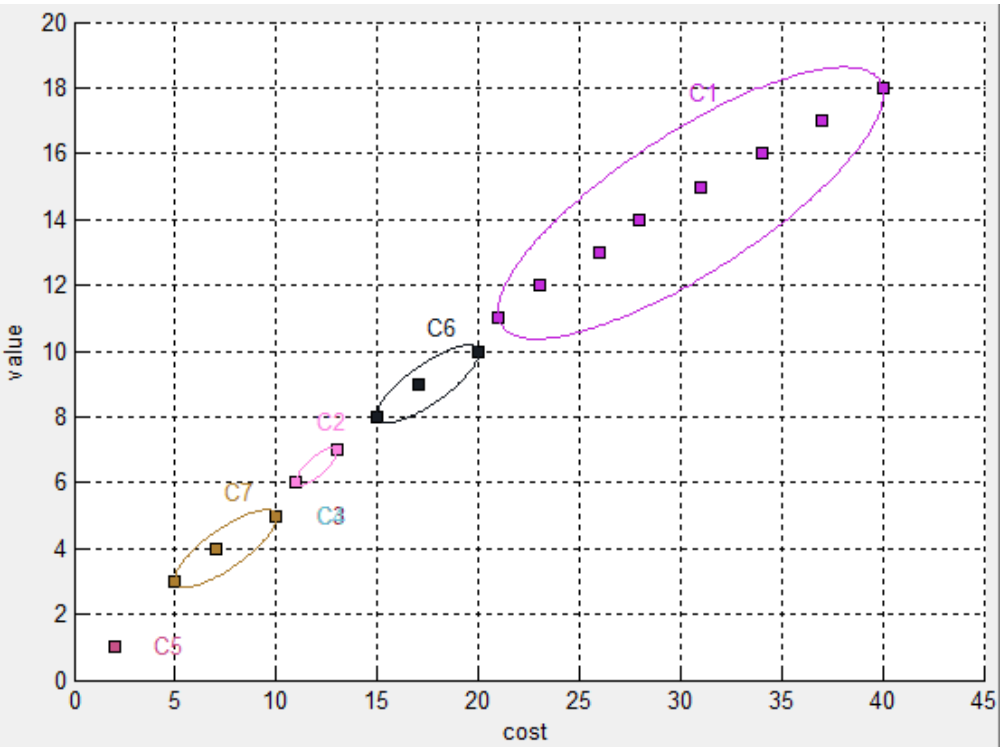


Figure 29 Cluster distribution on the Pareto Optimal Front

Ellipses that overlap on the Pareto front (i.e. that have non disjoint cost and value ranges) indicate an area where there may be solutions with very different requirements selections that achieve similar levels of objective attainment. The overlapping between C7, C4, and C3 is an example of this. Not all overlapping ellipses, however, will correspond to strongly different solutions. If the inter-cluster distance between two overlapping clusters is small, it is likely that the boundary between them is not so clear-cut and that solutions that belong to their intersection on the Pareto front are in fact close to each other in terms of selected requirements. Inspecting the requirements

distribution for each cluster (as supported by the views presented in the next section) allows decision makers to check whether this is the case or not.

Areas with adjacent but non-overlapping clusters may indicate points on the Pareto front where there are significant differences between solutions below and above a certain cost point. Such areas exist in Figure 29 notably between C7 and C2, C2 and C6, and C6 and C1. Again inspecting the requirements distribution within adjacent cluster will help understanding the relationship between them.

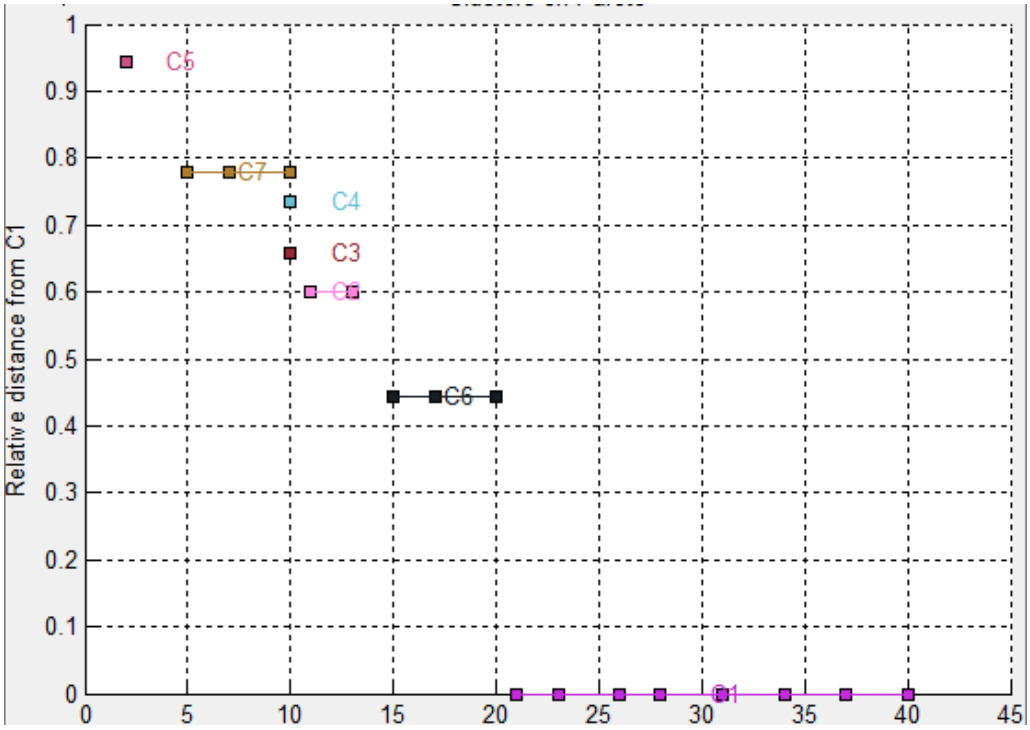


Figure 30 Alternate View of Clusters

Clusters with a single solution are also worth inspecting because they may denote solutions that are very different from all others around them on the Pareto curve. The clusters containing the single solutions S1, S5, and S7 are example of this.

On Figure 29, it is possible to zoom in on an area of a Pareto front to distinguish clusters more clearly when there are many clusters and many solutions in a small region. Alternatively, one can ask for clusters to be re-regenerated by taking into accounts only the solutions that are within some cost or value range. The regenerated clusters will in general be different from the first set of clusters. This can be used notably to focus on a region of the Pareto front that includes overlapping clusters to check whether solutions that were in different clusters in the first clustering remain in different clusters in the

second clustering. If this is the case, it would increase the chance of finding strongly different solutions within that region.

Details about each cluster's composition are also given by extending the solution table (such as Table 5) with a column indicating the solutions' cluster and the rows are coloured with the solutions' cluster's colour. Such tables can be ordered by clusters to inspect details of individual clusters. They can also be ordered by cost or by value which, with the help of the cluster colours, makes it easy to see cluster's overlap in the table.

5.3.2 Requirements Distribution per Clusters

The second set of views aims to help decision makers understand the compositions of each clusters and the relations between different clusters as cost increases. In these visualizations, the clusters are ordered by cost, with the cheapest one first and most expensive last. We also sort the requirements in increasing order of cost (with the cheapest at the top and the most expensive at the bottom).

Figure 31 shows the requirements distribution view for each cluster in the running example. Such view shows for each cluster a bar chart that gives for each requirement the percentage of solutions in the cluster in which it is selected. The requirements are sorted in order of increasing cost while the clusters are organized by increasing order of average cost and annotated with their size. This view can help visualizing whether overlapping or adjacent cluster are strongly different or not. For example, this view shows that clusters C7, C3, and C4 that overlap on the Pareto front have very different compositions. There are also significant differences between the adjacent clusters C2 and C6: all solutions in lower cost cluster C2 include R3, whereas none of the solutions in higher cost cluster C6 does.

A related view presents the same information as the bar charts but in a tabular form similar to a pH chart as shown in Figure 32. In this view, the distribution of each requirement within each cluster is now indicated by a colour scheme (the darker the cell, the more the requirement is present in solutions within the cluster). The ranges for each colour shade are described in a key table. The labels 'None', 'Some', and 'All' are also used to indicate whether the requirement is present in none, some, or all solutions within the cluster. Such view allows one to easily identify how the clusters' compositions evolve with cost and to identify which requirements tend to be present in lower cost, middle

cost and higher cost clusters. For example, all solutions in cluster 7 include R2, 75% of them include R7, 25% include R6, and 25% include R8

One interesting finding from both visualisations is the fact the C3 is a single solution cluster that includes only requirement R1 which is the most expensive one and it is between clusters which have more heterogeneous compositions.

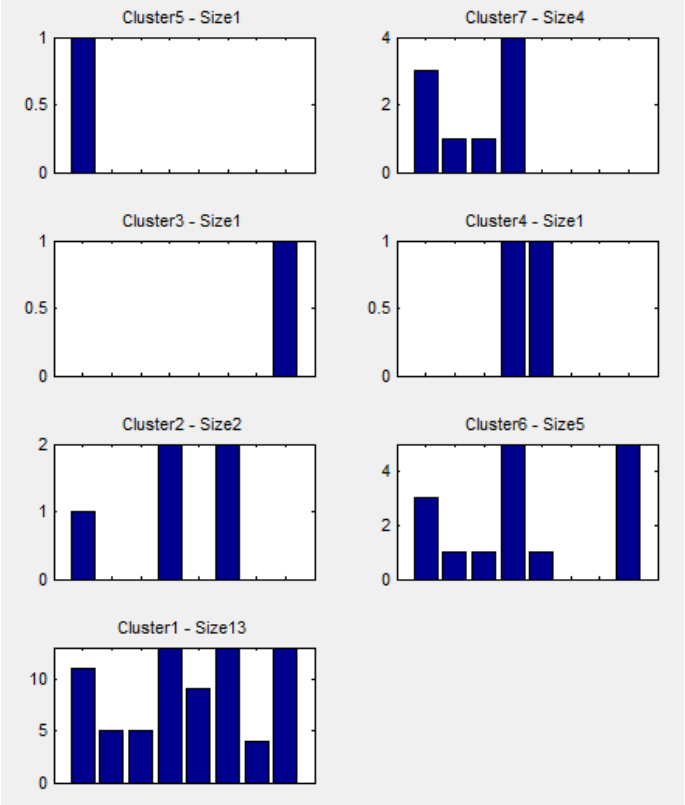


Figure 31 Requirements Distribution per cluster

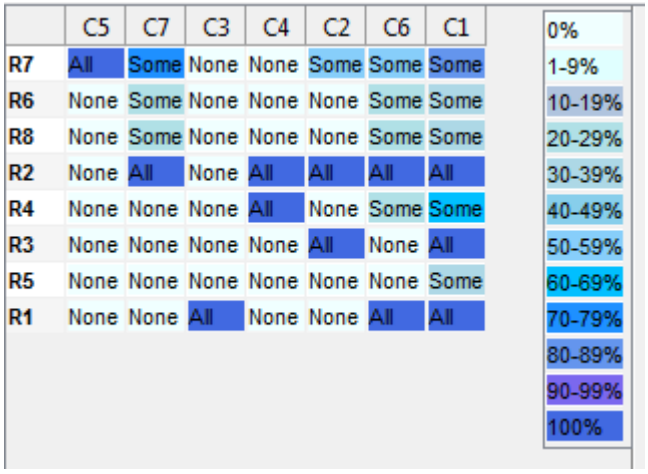


Figure 32 Cluster "Ph" chart

5.3.3 Pair-wise Comparison of Clusters' Compositions

We also found it useful to perform pair-wise clusters comparisons. The purpose of this view is to make it easy to identify what is common and what is different between solutions found in both clusters. For example, Figure 33 shows the pair-wise comparison between the adjacent clusters C2 and C6. This view helps highlighting which requirements are present in both solutions (R2), absent in both (R5), and which are present in all solutions of one and absent in all solutions or the other (R3 and R1).

This view helps decision makers verifying that the separation between adjacent clusters is meaningful, and it helps them understand the key differences between the two clusters (in this example, the highest cost solutions in C6 include R1 that is absent from solutions C2 but exclude R3 that is included in C2).

		C6		
		None	Some	All
C2	None	R5	R4 R6 R8	R1
	Some		R7	
	All	R3		R2

Figure 33 Pair-wise comparison of clusters

5.4 Tool Support for Optimal Solutions Analysis

We have developed a tool in Matlab to enable decision makers to perform *optimal solutions analysis* easily. It takes as input the output vectors generated from multi-objective search-based decision making techniques and performs cluster analysis on the solutions. The output is the dendrogram, the clusters of solutions and visualizations to enable further analysis of the clusters. The tool also enables the user to use the default cut-off value or manually choose a cut-off value to suit his needs when choosing clusters.

This tool is further described in Appendix B.

5.5 Related Work

The idea of clustering solutions in a Pareto optimal set is not new (Morse 1980; Rosenman and Gero 1985; Mattson, Mullur, and Messac 2004). The approach taken by previous techniques is to cluster solutions according to how close they are in term of objective attainment (which is useful to help understanding solutions when the number of objective is larger than 3); our approach in contrast is to cluster solutions according to

how close they are in terms of design decisions. This latter approach has also been proposed to help understanding optimal solutions in industrial design problems (such as optimizing the dimensions of a combustion engine's exhaust pipe) (Aittokoski, Ayramo, and Miettinen 2009). The design decisions in such problems consist in selecting optimal values for a small number of continuous variables. In contrast, the design decisions for requirements selection problems consists in making decisions for a large number of Boolean variables (indicating whether a requirement is selected or not). The difference is significant as it requires entirely different specification of distance functions, different clustering approaches, and different cluster visualizations.

5.6 Conclusion

Many decision problems in requirements engineering, such as the cost-value based requirements selection problem and NASA's DDP (Feather and Menzies 2002) risk mitigation selection problem, rely on quantitative multi-objective decision techniques and search-based algorithms to generate sets of optimal solutions. These sets are usually analysed in the objective space (by visualising the Pareto front curve) to inform possible trade-offs between conflicting objectives. Little work has been done so far to support decision makers in understanding variations between solutions in the design space (i.e. how they vary in terms of selected requirements).

We have seen that identifying groups of strongly related solutions may improve the quality and ease of the decision making process by (1) helping decision makers in understanding how groups of solutions are spread on the Pareto front, (2) helping them in identifying areas where strongly divergent solutions achieve similar objectives (which can notably be important when planning for potential extension and contraction of a solution), and (3) allowing them to make decisions incrementally by first selecting within groups of solutions before selecting one of the variants within the chosen group.

We have proposed a hierarchical clustering technique relying on a weighted distance function as an appropriate technique to group solutions for requirements selection problems. We have then proposed a series of visualizations to support decision makers achieving the three goals mentioned in the previous paragraph.

Chapter 6 - Validation

In this chapter, we will be looking at how the techniques proposed fare in real world projects.

6.1 Introduction

We next validate our proposed techniques using two case studies. We look at how our framework contributes towards improving the requirements selection and optimisation process.

For the *stakeholders' preferences analysis* technique, we check whether our approach allows us to form stakeholder clusters that have preference values closer to the actual individual preferences of the stakeholders than other approaches. We also verify if our approach allows us to identify trends in the preferences of the stakeholders as well as 'outliers' stakeholders in the case studies.

For the *optimal solutions analysis* technique, we check whether our technique helps in identifying and understanding variations within the set of optimal solutions by grouping them according to their design similarities.

We use our tool to perform the validation on the datasets we have identified. In each case, the tool computes the Cophenetic Correlation coefficients and automatically chooses the linkage that is more appropriate for the data. Similarly, our tool also determines the Mojena cut-off and uses this as the default cut-off.

We have recorded the steps in using the tool support for both the *stakeholders' preferences analysis* and *optimal solutions analysis* for the RALIC dataset and supplied the video as a demo of how to use the tools. These videos, the installation files of the tools and the artificial running example datasets are available online[9].

[9] http://www.veerappa.net/tools/

6.2 Case Studies

We have identified two datasets to use to validate our proposed techniques. These are the Replacement Access, Library and ID Card project (RALIC) dataset (Lim 2010) and an industrial Motorola dataset (Baker et al. 2006).

6.2.1 RALIC Case Study

Previously, UCL had numerous access and security systems controlling access and identification specific to each facility. Consequently, members of staff, students and visitors had to use at least two access control systems. These included the Photo ID Card, the Library Barcode, Session Card and Bloomsbury Fitness Centre Card among others. The RALIC project is a large scale software project at the University College London which aims to centralise access control to the different facilities available at UCL into a single card. The objectives of RALIC includes the replacement of magnetic swipe card readers with smart card readers, the definition of user groups and default access rights and the replacement of the Library Access Control system (Lim 2010).

The project involves more than 60 stakeholder groups that have been identified during the stakeholder analysis phase and about 30000 regular users of the system to access UCL building, the library resources and IT facilities. The identified stakeholder groups include students, academic staff, academic visitors, security staff, developers, managers and administrators from academic departments and staff from the Information Services Division. Data about this project was collected via the StakeSource (Lim 2010) platform. An initial set of stakeholders were asked to recommend other stakeholders who they think were important for the project.

Of these stakeholders, 87 agreed to provide their full profile and they were retained for the project. The next step involved rating and discovering requirements. The stakeholders were presented with a list of initially identified requirements and were asked to rate them on a scale of 0-5 where 0 meant that they did not care about the requirement and 5 meant that the requirement was of utmost importance to them. Furthermore, they were also provided with the option to rate a requirement with a value of -1 if they did not actively want the requirement to be included in the system. The stakeholders could also define new requirements and these were made available for rating purposes to the others as far as possible.

Among the data collected, the ratings elicited from 76 stakeholders on 10 project objectives, 48 requirements and 104 specific requirements are of relevance in our context. We will consider only 99 of these requirements for the validation. The cost data are obtained from the RALIC post implementation report. The cost of each requirement is the time, in terms of person hours, spent by the project team on the requirement during the project. For requirements that were not implemented, the cost is estimated by inferring from the cost of similar implemented requirements. This estimated cost has been ratified by the project team. Table 10 shows 23 requirements and their corresponding costs (Lim 2010).

ID	Requirement	Cost (person hours)
a.2	use the same access control for library entrance	411
a.3	all in 1 card	
a.3.1	combine ID card and session card	158
a.3.2	combine Library card	276
a.3.3	combine Bloomsbury fitness card	189
a.3.4	combine Club and societies card	76
a.3.5	be compatible with NHS	76
a.3.6	the combine card should not have too many features (don't want it to become too valuable to change for locker keys)	9
b	card design	
b.1	card to include user details	
b.1.1	card to include name	4
b.1.2	card to include photo	8
b.1.3	card to include UPI	4
b.1.4	card design to include card type/user status	11
b.1.5	card to include payroll number	24
b.1.6	card to include job title	8
b.1.7	card to include expiry date	4
b.1.8	card to include department	57
b.1.9	card to include student number	13
g.2	export data to other systems	
g.2.1	export data to student system	291
g.2.2	export data to library (access card changes, leavers, barcode)	189
g.2.3	export data to staff system	252

Table 10 Examples of RALIC Requirements and their Costs

We use this data to validate both the *stakeholders' preferences analysis* and the *optimal solutions analysis* techniques. Firstly, we cluster the stakeholders according to their

preferences to get initial stakeholder groups to feed into the simple NSGA-II search-based requirements prioritisation technique provided by Zhang (Zhang 2010) with the objective of maximizing value while minimizing cost. We then cluster the solutions on the resulting Pareto Optimal front to perform further analysis on the possible design trade-offs that can be made.

We assume here there are no dependencies (or requirements interaction constraints) among the requirements when performing the search and use the NSGA without dependencies algorithm(Zhang and Harman 2010). However, Zhang et al have shown that it is possible to use requirements interaction constraints when generating the Pareto front for this dataset (Zhang, Harman, and Lim 2012). The output in the latter case could also be used as input to our technique.

It has been observed that for some of the requirements, the stakeholders did not provide any value. Since we are first using the *stakeholders' preferences analysis* technique on this case study, these missing preferences will be represented by the cluster preference for the cluster in which these stakeholders are. We specify a value zero (the don't care value in this case) for those requirements for which our technique could not find a cluster preference.

6.2.2 Motorola Dataset

The other case study that we use for evaluation purposes is the Motorola dataset (Baker et al. 2006) which was elicited during a project where Motorola was designing a new communication handset. The stakeholders are four mobile telephony service providers. They were asked to rate features they believe should be included in the new handset. As expected, they each had specific sets of priorities in this respect. Thirty five requirements were identified by Motorola and the estimated cost of implementation was used as the cost of implementation. We are using the anonymised version of this dataset previously used for validation of other techniques (Baker et al. 2006). We use this dataset to validate only the *optimal solutions analysis* technique. This is because our *stakeholders' preferences analysis* only works for large numbers of stakeholders. The Motorola dataset only has 4 stakeholders involved making it unsuitable for this technique.

6.3 Clustering Stakeholders for RALIC

We first apply our *stakeholders' preferences analysis* technique to cluster stakeholders on the RALIC case study. The requirements have been coded *a*, *a.1*, *a.1.1* and so on. This coding convention has been derived as follows: the letter represents a system objective; the numbers represent requirements in a hierarchical structure. Thus for objective *a*, we will have high level requirement *a.1* and a lower level requirement *a.1.1* which is sub-requirement of *a.1*.

The structure of the hierarchy is depicted in Figure 34. In our work, we will be looking at the ratings for leaf requirements of the hierarchy – these are the requirements that are bottom of the hierarchy. There are 99 such leaf requirements.

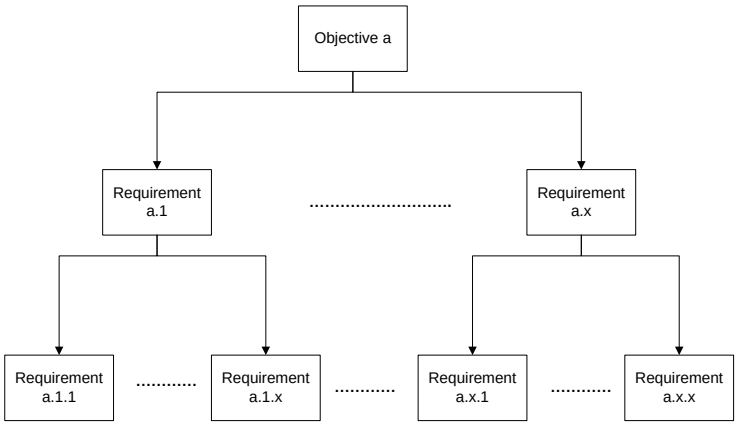


Figure 34 Hierarchical structure of RALIC requirements

For the RALIC project, 23 groups of stakeholder roles have been identified. These include divisions such as the Security and Access Systems, positions such as Heads of Departments, and groups like Students and organisations like the card vendor (Lim 2010). All stakeholders belong to one or more of these groups, except 6 stakeholders which have thus been assigned to a group "not available" which we label as "na". These groups and the total number of stakeholders in each of them are listed in Table 11.

For the purpose of our validation exercise we have formed 8 higher levels of stakeholder groups to have a more significant population for each group. The stakeholder groups are: admin, technical, academic, student, other staff, security staff, potential criminals and we have kept the group "na" for stakeholders for which we have not been able to retrieve any role related information. For example, stakeholders from the Information Services Division have been assigned to the group "technical". We have a

group "potential criminals" which consists of a security expert identifying which requirements are useful to stop potential criminals. The new distribution of stakeholders is shown in Table 12.

Since this requirements decision process concerns UCL security systems, we want our clusters to be of good quality to ensure that our decisions will reflect the needs of the stakeholders as realistically as possible. We will thus be checking the validity of the clusters before recommending groups to be used in the decision process using the Rand, C and Silhouette indices mentioned in Chapter 4.

Role	Total
Clubs and Societies	1
Corporate Support Services	1
Dean of Students	1
Departmental Administrators	1
Disability Centre	1
Estates and Facilities Division	6
gym users	2
head of departments	2
Human Resources Division	3
Information Services Division	1
Information Strategy Committee, Information Services Division	1
IS	11
Library Services	2
Management Systems	15
NA	6
potential criminals	1
Registry	4
Security and Access Systems	3
Security Staff	3
Senior Tutor	1
Staff	4
Students	2
UCL Union	3
Web Services	1

Table 11 Roles of Stakeholders

Group	Total
Academic	5
Admin	21
Criminals	1
NA	6
Other Staff	1
Security Staff	5
Student	4
Technical	33
Total	76

Table 12 Groups of Stakeholders

We first use the *stakeholders' preferences analysis* technique to find the preference-based clusters for the RALIC case study. The cluster preference is chosen as the median value of the ratings. We compare how well the preference value our technique generates performs against the usual overall preference where all stakeholders are viewed as a single grouping and the group preferences for product-independent stakeholder groups. We compute the distances $divergence_o$, $divergence_g$ and $divergence_c$ defined in Chapter 4 and view their box plots to visualize how these divergences vary. Finally we try to uncover any patterns or exceptions in the preferences of the clusters using statistical analyses.

We prepare the data gathered into an Excel sheet as required by the tool (Appendix A) to be able to run our technique on it. The tool first generates the dendrogram for the data as shown in Figure 35. It also computes the default cut-off value using the Mojena cut-off value depicted as a black dotted line on the dendrogram.

6.3.1 Observations on the Dendrogram

We can see that the data tends to form two big clusters (coloured yellow and red on the dendrogram) as well as two clusters with only one stakeholder, one of which is merged very late in the clustering process. The default cut-off here generates 9 clusters. Since we need to use clusters which are as valid as possible, we check the quality indices for different cut-offs from the graphs generated by the tool as shown in Figure 36. We can see that the default cut-off is a good choice as it has good Rand, Silhouette and C indices.

6.3.2 Analysis of Clusters

Our technique enables requirement decision makers to further investigate the clusters to determine their composition and detect any interesting trends. We next look at the different analyses we can do on the clusters generated and identify any interesting trends from these.

Clusters Summary

In this view, we can see information such as the size of the clusters and the representative rating values for each of the requirements for the individual clusters. This information is presented in tabular form on the tool by default. We can see from the tabular information about the clusters in Table 13 that we have 2 major clusters C7 and

C8 with sizes 17 and 38 respectively. There are 2 single-stakeholder clusters and the remaining 5 stakeholder clusters have between 5 and 3 stakeholders. The complete detailed cluster composition for the RALIC stakeholders can be found in Appendix C. We are only showing 8 requirements here.

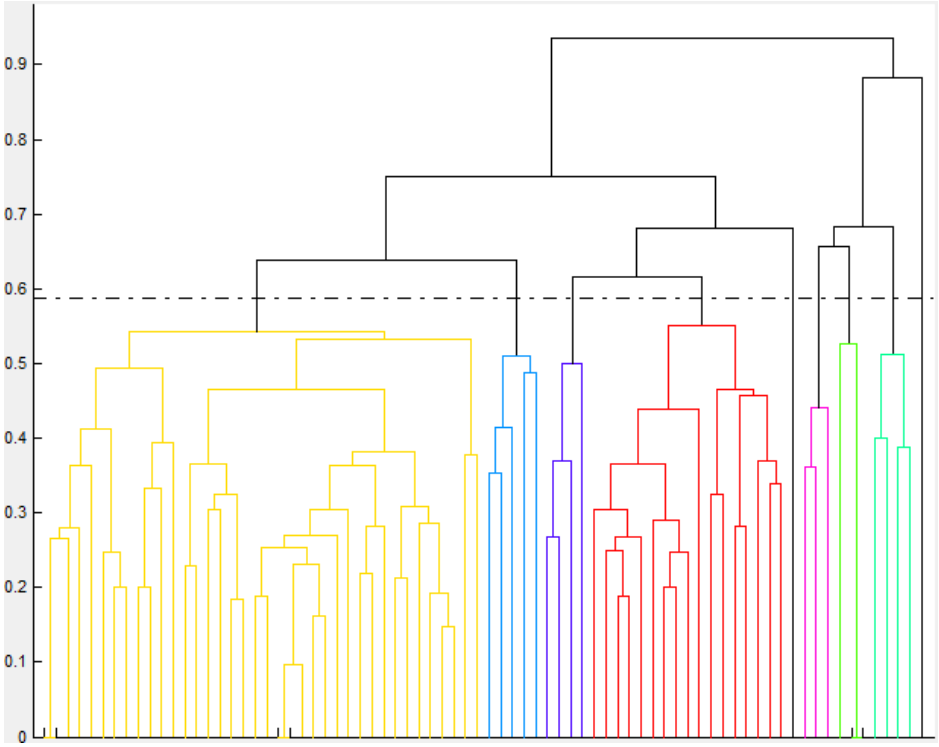


Figure 35 Dendrogram for Stakeholder Clusters in the RALIC Case Study

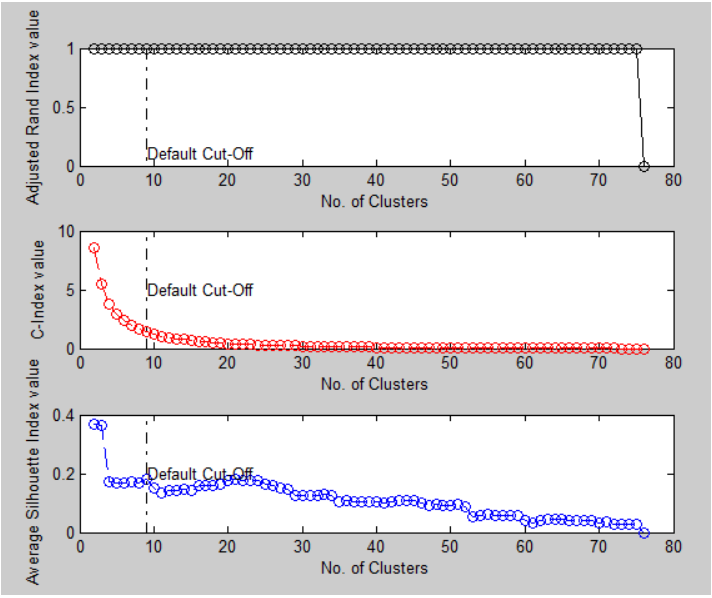


Figure 36 Quality indices for Stakeholder Clusters RALIC Case Study

Cluster	Size	a.1.2	a.1.3	a.2	a.3.1	a.3.2	a.3.3	a.3.4	a.3.6
1	**4**	4	4	4	4	4	4	4	4
2	**5**	5	5	5	5	5	5	5	5
3	**3**	3	3	2	4	4	4	4	4
4	**1**	0	0	0	0	0	0	0	0
5	**4**	1	1	1	1	1	1	1	1
6	**1**	5	5	5	4	4	-1	0	
7	**17**	4	4	4	5	4	4	4	4
8	**38**	5	5	5	5	5	5	5	5
9	**3**	3	3	1	5	5	5	5	5

Table 13 Cluster Summary for RALIC Stakeholders Clusters

Our tool enables the decision maker to export this cluster summary table into Excel. When doing so, if there are requirements for which no median value has been found (for example requirement a.3.6 for cluster 6), our tool asks the decision maker to decision for a value to use in these cases.

Analysing clusters of stakeholders

The detailed cluster composition in Table 14 enables us to identify which stakeholders are in which clusters. We have shown only the first 8 requirements for 40 stakeholders here (all the names anonymised). For example, we can see who are in the single-stakeholder clusters and therefore who are outliers.

For clusters with more than one stakeholder, our technique allows the requirements decision maker to further analyse how the ratings vary within the clusters by plotting the box plot of the ratings for all the stakeholders in that cluster for all requirements. The box plot for three largest clusters C2, C7 and C8 are shown in Figure 37. The requirements have been sorted in increasing cost.

One of the information on the box plot of ratings is the fact that cluster 8 consists of stakeholders who have given a rating of 5 to the low cost requirements *c.2.4, c.2.7, c.2.9, c.2.10, c.4.3* and *a.3.6*. It is also clear that out of the 38 stakeholders in C8, all have given a rating of 5 to these requirements except for a few outliers. For these same requirements, we can see that cluster 2 shows a greater variation in the preferences of the stakeholders. 50% of the stakeholders have a preference between 2 and 4 for *c.2.4, c.2.7, c.2.9* and *c.2.10* while for *c.4.3* the preferences vary between 1.5 and 5. In C7, however, we can see that the stakeholders have mostly a preference of 4 for these requirements. Another observation is that the stakeholders in cluster 2 do not care about requirement *b.1.4* but

all stakeholders in cluster 7 have a preference of 4 for it while those in cluster 8 have a more varied preference with 50% of them rating it between 3.5 and 5.

Cluster	a.1.2	a.1.3	a.2	a.3.1	a.3.2	a.3.3	a.3.4	a.3.6	Stakeholder
1	4	4	4	4	4	4	4	4	Tony Boston
1									Brian Ward
1	3	3	3	4	4	4	4	4	Oliver Cullen
1	5	5	5	5	5	5	5	5	Ruth Simon
2	5	5	5	5	5	5	5	5	Magali Persi
2	5	5	4	5	5	4	4	4	Majid Khande
2	5	5	5	5	5	5	5	5	Bob Alford
2	5	5	4	5	5	5	5	5	Roger All
2	5	5	5	5	5	5	5	5	Shawn Wills
3	3	3	3	4	4	4	4	4	Alison Crane
3	5	5	0	3	3	3	3	3	Andy Kirb
3	2	2	2	4	4	4	4	4	Samuel Mackey
4	0	0	0	0	0	0	0	0	Conrad Moore
5	1	1	1	1	1	1	1	1	Angela Willard
5	1	1	1	1	1	1	1	1	David Carne
5	2	2	2	2	2	2	2	2	Sean Wall
5	0	0	0	0	0	0	0	0	Tim Pugh
6	5	5	5	4	4	-1	0		Pepi Sands
7	4	4	4	4	4	4	4	4	Adrian Bank
7	5	5	4	5	5	5	5	5	Andy Faulk
7	4	4	4	5	4	4	4	4	Brian Aniston
7	3	3	3	4	4	3	3	3	Caroline Goodman
7	3	3	3	3	3	3	3	3	Colin Penn
7	4	4	4	5	4	4	4	4	David Ainsley
7	5	5	5	5	5	5	5	5	Liz Hopper
7	4	4	4	4	4	4	4	4	Fickle Andrews
7	5	5	3	5	3	3	3	3	Jason Ortiz
7	5	5	5	5	5	5	5	5	Johnny Glenn
7	3	3	4	5	5	5	5	5	Kathryn Lester
7	5	5	5	5	5	5	5	5	Marilyn Gallo
7	2	2	2	4	4	4	4	4	Mark Wesley
7	5	5	5	5	5	5	5	5	Noshir Holmes
8	5	5	5	5	5	5	5	5	Niyi Akers
8	4	4	4	5	5	5	5	5	Aaron Toms
8									Andrew Dawn
8	5	5	5	5	5	5	5	5	Astrid Haynes
8	5	5	5	5	5	5	5	4	Barbara Song
9	2	2	1	5	5	5	5	5	Andy Hicks
9									Christina Solis
9	4	4	1	5	5	5	5	5	Simon Farmer

Table 14 Detailed Stakeholder Cluster Composition for RALIC

One conclusion based on these observations is that the stakeholders from cluster 7 and 8 have a consistent high preferences for requirements *c.2.4, c.2.7, c.2.9, c.2.10, c.4.3* and *a.3.6* which have relatively low cost .

When comparing the box plots of the preferences of the stakeholder clusters to the box plot preferences of all stakeholders in Figure 38, we can see that for most of the requirements, the spread of the preferences has decreased indicating that stakeholders in a cluster have very similar preferences.

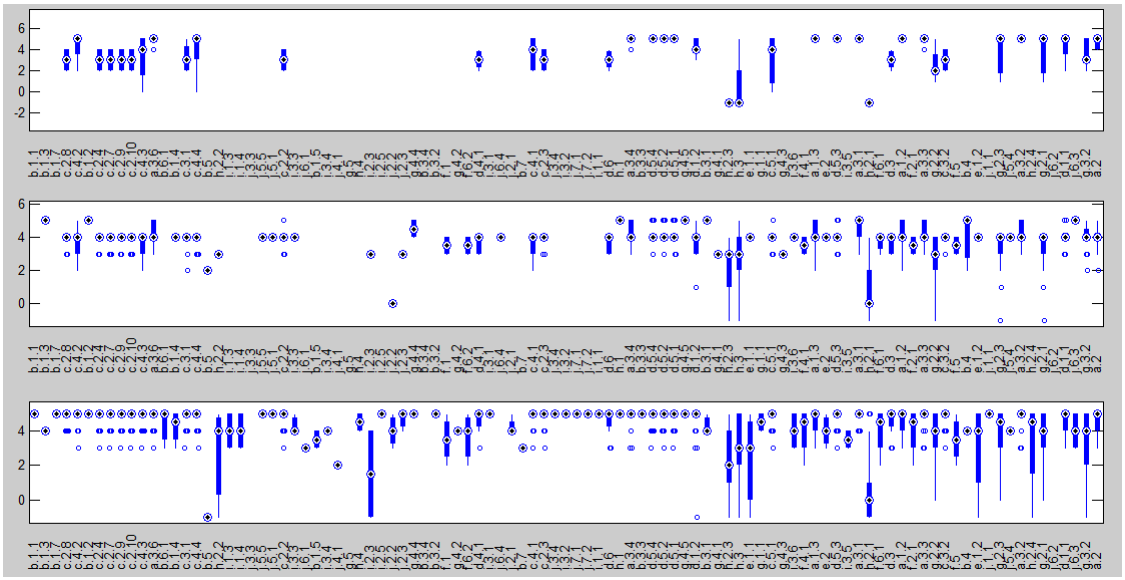


Figure 37 Box plot for preferences for Stakeholder Clusters C2, C7 and C8

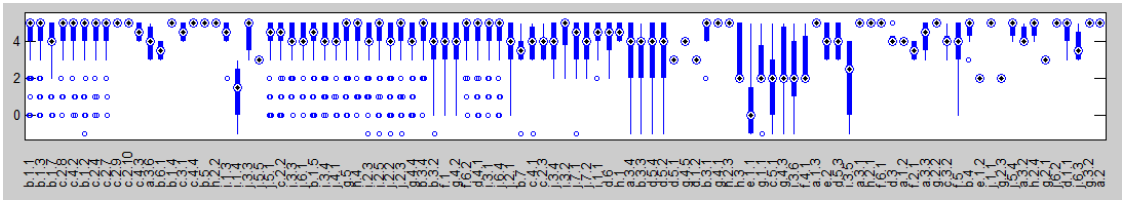


Figure 38 Box plot individual preferences for all stakeholders

We have used the cluster composition information from our technique to do some further manual analysis to understand their composition. Since we have the roles of the stakeholders, we can check the composition of the clusters by role. We have used pie charts to depict the distribution of stakeholders' roles by cluster and distribution of clusters among the stakeholders' roles.

Figure 39 and Figure 40 show us that the stakeholders for the two largest stakeholder groups are scattered among many stakeholder clusters. "Technical" staff is present in 8 clusters including single-stakeholder cluster 4. "Admin" staff, on the other hand, is present in only 5 clusters. The largest proportions of these stakeholders are in the largest clusters 7 and 8.

From Figure 41, we can see that cluster 8 consists of a more diverse population with stakeholder from all roles with admin and technical consisting of 79% of the cluster. However, cluster 1 consists of 2 "security staff" stakeholders and 1 "technical" stakeholder (Figure 42). From our previous analysis we have found that this cluster has negative preferences for requirements *g.2.1, g.2.2* and *g.2.3* which are related to exporting data. Exporting data in such systems can have security implications. This may explain why staff with a technical and security background are unwilling to have this requirement in the system. The "admin" stakeholder in this cluster seems to be the odd member of this cluster. The decision maker may want to further investigate why this stakeholder has voted against exporting data.

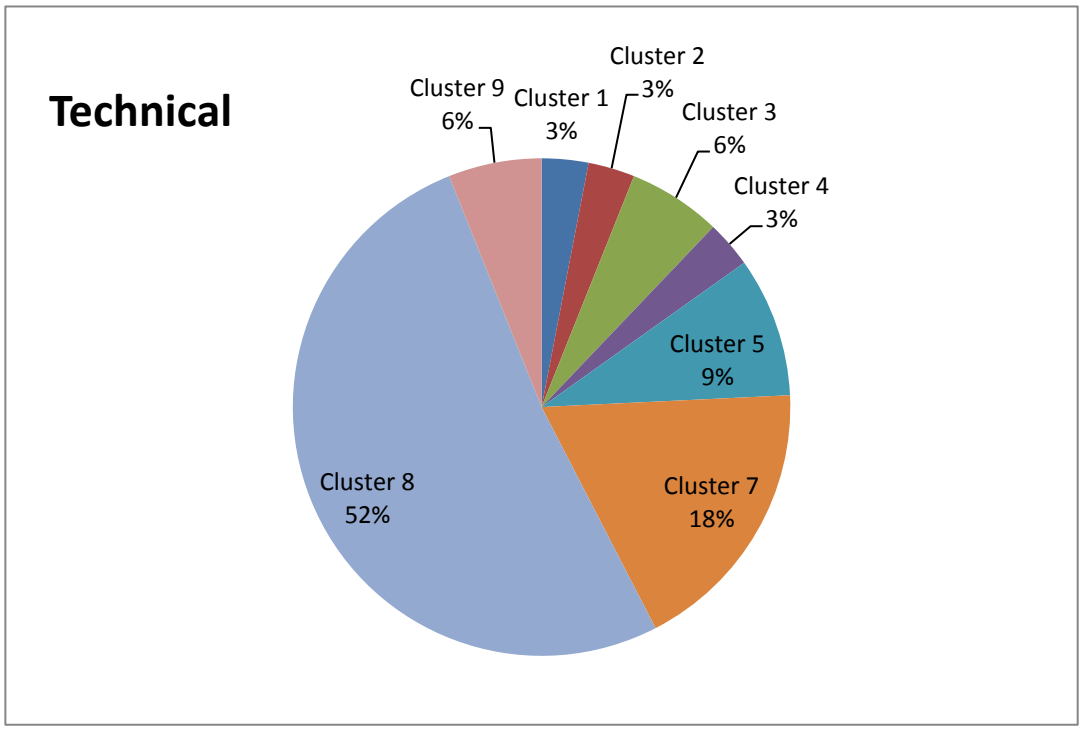


Figure 39 Technical Distribution per Cluster

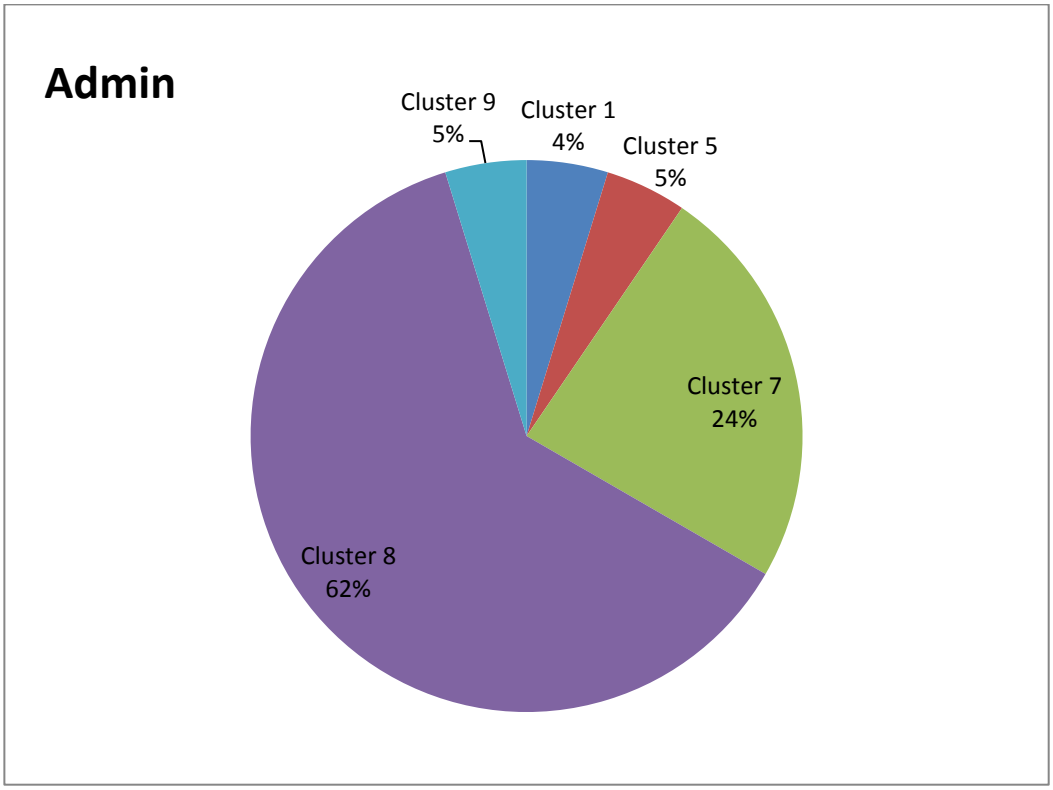


Figure 40 Admin Distribution per Cluster

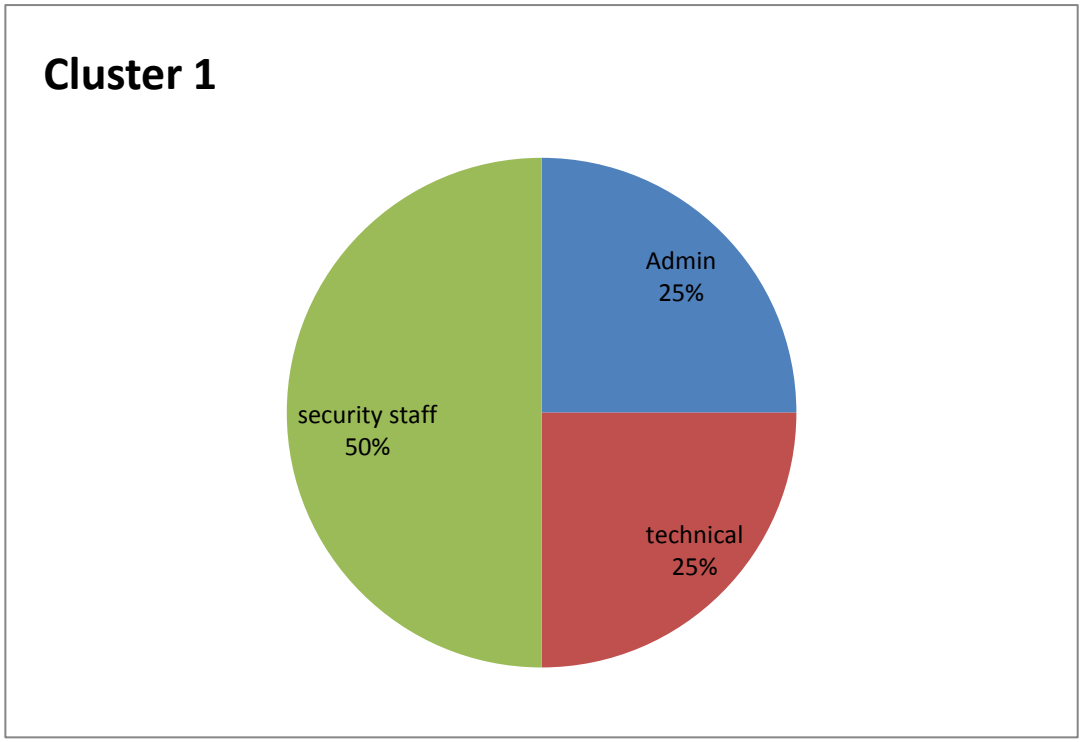


Figure 41 Cluster 1 Distribution per Role

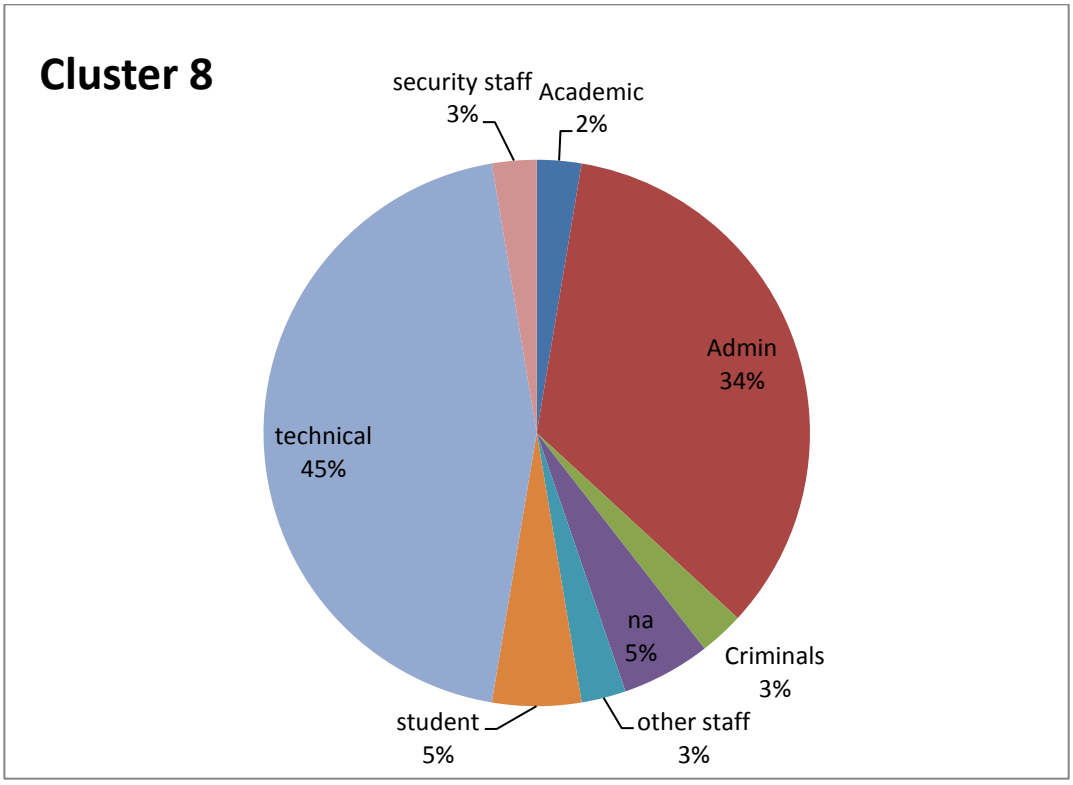


Figure 42 Cluster 8 Distribution per Role

Analysing individual stakeholders

For the RALIC case study, one major observation is that we have 2 clusters with one stakeholder in them. Depending on the context of the project, we may or may not want to use all generated clusters, including single-stakeholder ones. As mentioned in Chapter 4, if an individual stakeholder has a high importance in the project, we may want to include him/her in our decision making process. This involves going back to the stakeholder and understanding his preferences for the requirements. If the stakeholder cannot be contacted, his importance in the project can be used as the only criteria on whether to include him or not.

For the RALIC case study, we can use the pagerank (Lim 2010; Page et al. 1998; Brin and Page 1998) which measures the ranks a given stakeholder according to his relative importance with respect to all other stakeholders. This measure propagates importance as stakeholders who are highly recommended by many important stakeholders are important, and the recommendations of highly important stakeholders are given more weight, making their recommended stakeholders important. This data has already been

collected during the elicitation phase. Since we cannot contact the RALIC stakeholders, we can base ourselves on their importance to decide whether to include them or not.

We identify the individual 'outlier' stakeholders and their pagerank, and their resulting rank (from the pagerank value) in the group as listed in Table 15.

Name	Cluster	Rank	Pagerank
Conrad Moore	4	54	0.004685637
Pepi Sands	6	71	0

Table 15 Overall Rank and Pagerank of individual Stakeholders

From the rankings, we can ignore those stakeholders as they are with low importance in the project.

6.3.3 Evaluation of Preference Values

To evaluate how well the cluster preference represents the stakeholder in the decision, we determine if the *cluster preferences* we generate with our technique are closer to the *individual preferences* than the *group preferences* and the *overall preference*.

Overall Preference for RALIC Case Study

In this case, we assume that the value to be used to represent the stakeholders in the decision making process is obtained by computing the median of the ratings for each of the requirements. The representative values for the average stakeholder for the requirements *a.1.2* to *b.1.1* for the RALIC project are listed in Table 16.

Requirement	a.1.2	a.1.3	a.2	a.3.1	a.3.2	a.3.3	a.3.4	a.3.6	b.1.1
Median	5	5	4	5	5	5	5	5	5

Table 16 Overall Preference for RALIC Data for Requirements a.1.2 to b.1.1

Representative value for the product-independent stakeholder groups

We compute the representative value for each of the requirements for each of the stakeholder groups that we have identified for this case study.

In this step, we proceed by computing the median of the ratings of the stakeholder in each group for all the 99 requirements. The resulting median for the requirements *a.1.2* to *b.1.1* are shown for the 8 groups of stakeholders are shown in Table 17.

	a.1.2	a.1.3	a.2	a.3.1	a.3.2	a.3.3	a.3.4	a.3.6	b.1.1
academic	5	5	4	5	5	5	5	5	
admin	5	5	5	5	5	5	5	5	5
criminals	5	5	5	5	5	5	5	5	
na	5	5	5	5	5	5	5	5	
other staff	5	5	5	5	5	5	5	5	
security staff	4	4	3	4	4	4	4	4	
student	4.5	4.5	4	5	5	5	5	5	
technical	4	4	4	5	5	5	5	5	

Table 17 Group Preference for RALIC data from Requirements a.1.2 to b.1.1

The box plot of divergences in Figure 43 indicates that the divergences between the *individual preferences* of the stakeholders and the *cluster preferences* ($divergence_C$) have a lower median and smaller spread than in the case of the *group preferences* ($divergence_g$) and the *overall preference* ($divergence_o$). A lower median implies that the cluster ratings are closer to the individual ratings than with the other two metrics. A smaller spread means that range of the ratings within the cluster groups is much smaller than with the other two approaches.

Our claim is that is preferable to use the cluster preferences for decision making because these are closer to the individual preferences than the alternative approaches of grouping stakeholders by product-independent characteristics or using a single large groups of stakeholders.

6.3.4 Conclusion

We have shown that the value for stakeholder preference being fed to decision-making techniques is closer to the actual preference of the stakeholder when we perform our *stakeholders' preferences analysis* first. We have also been able to show that our techniques enables decision makers to further identify trends in the preferences of stakeholders and possible "outlier" stakeholders with very diverging preferences. This information can be very useful to further understand the needs of the stakeholders and identify possible aspects of the requirements that the requirements engineers may have overlooked during the elicitation phase.

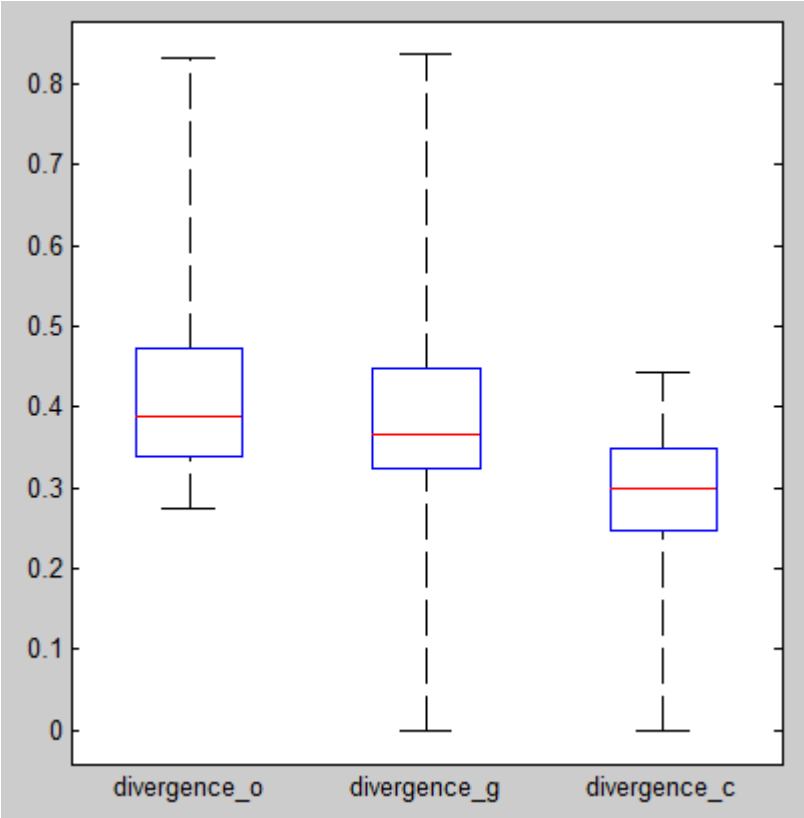


Figure 43 Box plot of distances of divergences for the RALIC case study

We next evaluate how clustering solutions on the Pareto Front help requirements decision makers to determine similarities and differences on the Pareto Optimal solution. We use both the RALIC and the Motorola datasets.

6.4 Clustering Pareto Optimal Solutions for RALIC

In this section, we evaluate if clustering solutions on the Pareto front from search-based requirements prioritisation helps to identify similarities and differences on the front. We use the clusters of stakeholders obtained in section 6.3 to feed into the simple NSGA-II search-based requirement prioritisation and selection technique described in Chapter 2 to generate the Pareto front. We ignore the two single-stakeholder clusters 4 and 6 when generating the Pareto front.

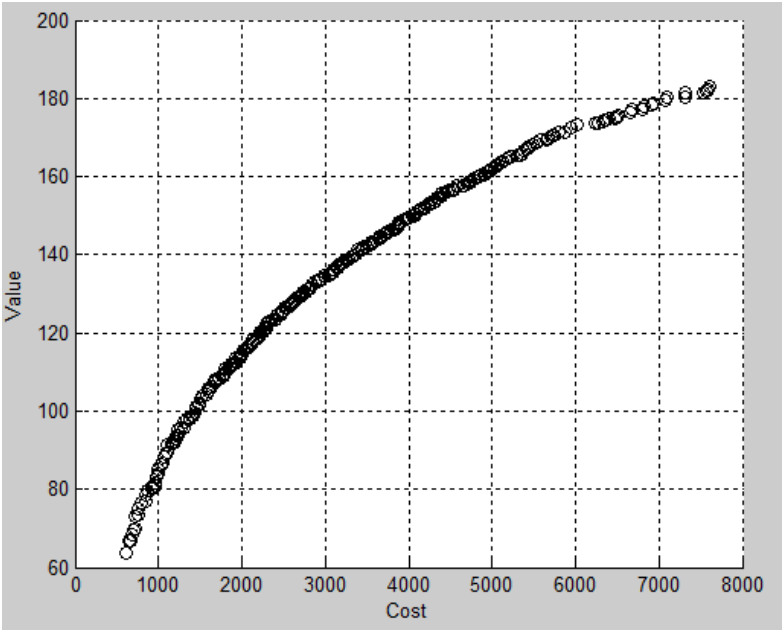


Figure 44 Pareto Optimal Front for the RALIC Case Study

The cluster preferences for the 7 selected clusters and cost for the requirements are used as input to the search-based algorithm which generates the Pareto optimal front depicted in Figure 44. There are 274 possible solutions spanning from cost 608 to 7306 and value from 63.79 to 180.29.

We next cluster the results using our *optimal solutions analysis* technique to identify similarities and differences among the solutions on the Pareto front and use the different visualisations provided in our tool to make necessary analyses.

Figure 45 shows the resulting full dendrogram with the default cut-off when the hierarchical clustering algorithm is executed on the data. We have weighted the distance by cost in this case. We keep the default cut-off as it has good Rand, C and Silhouette

indices as shown in Figure 46. The resulting 26 clusters are displayed along the Pareto front in Figure 47.

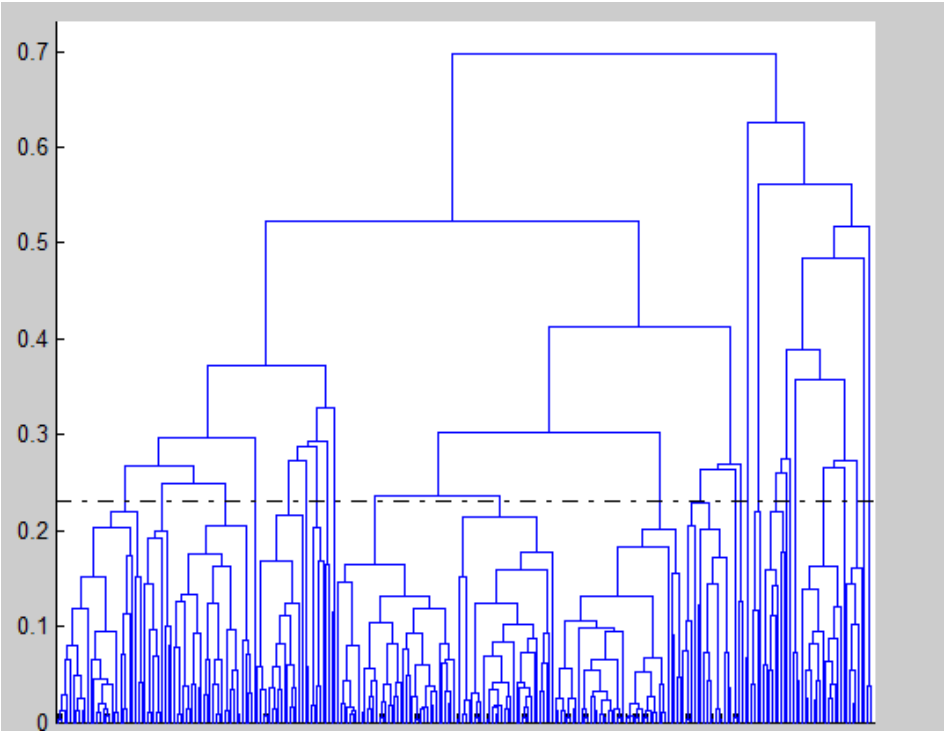


Figure 45 Dendrogram for RALIC solutions

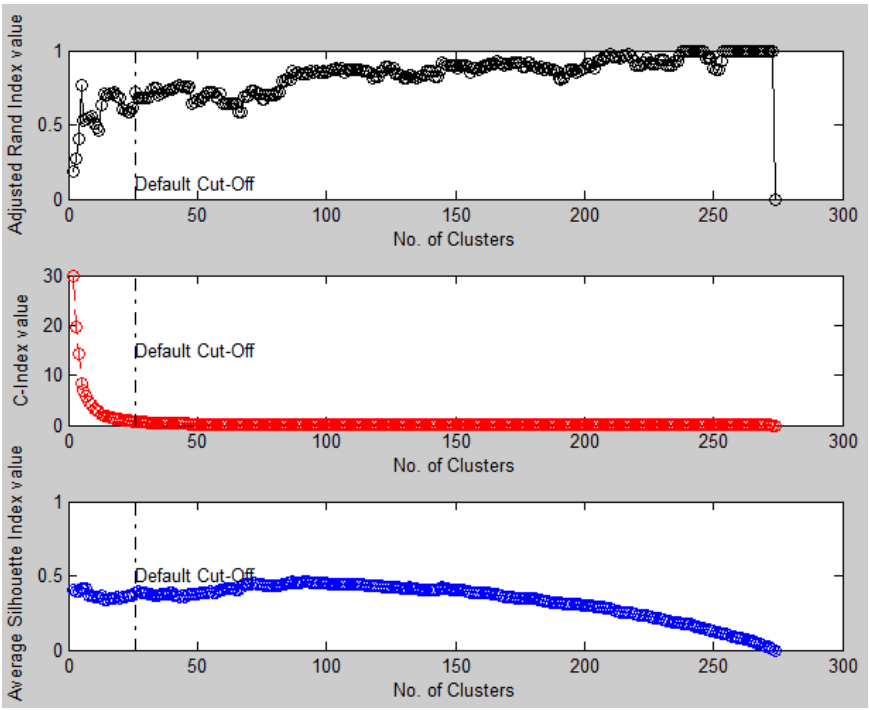


Figure 46 Quality indices for RALIC solutions

Our technique generates the cluster summary table for the clusters as well. This composition table is a 274 × 103 matrix in this case. Table 18 shows the cluster summary information for 19 solutions for the first 10 requirements.

Cluster	Solution	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	Value	Cost
C1	**S134**	1	1	0	1	1	1	1	1	1	1	135.07	3057
C1	**S135**	1	1	0	1	1	1	1	1	1	1	135.07	3067
C1	**S142**	1	1	0	1	1	1	1	1	1	1	137.64	3188
C2	**S115**	1	1	0	1	0	1	1	1	1	1	128.00	2616
C2	**S116**	1	1	0	1	0	1	1	1	1	1	128.29	2642
C3	**S17**	0	0	0	0	0	0	1	1	1	1	80.71	957
C4	**S256**	1	1	0	1	1	1	1	1	1	1	174.50	6374
C4	**S257**	1	1	0	1	1	1	1	1	1	1	174.79	6400
C5	**S42**	0	0	0	0	0	0	1	1	1	1	98.71	1397
C6	**S72**	0	1	0	1	0	0	1	1	1	1	112.64	1923
C6	**S77**	0	1	0	1	0	0	1	1	1	1	114.50	1996
C7	**S261**	1	1	1	1	1	1	1	1	1	1	176.21	6662
C7	**S269**	1	1	1	1	1	1	1	1	1	1	180.29	7306
C8	**S31**	0	0	0	0	0	0	1	1	1	1	91.71	1173
C8	**S34**	0	0	0	0	0	0	1	1	1	1	93.64	1217
C9	**S64**	0	0	0	1	0	0	1	1	1	1	109.21	1789
C9	**S70**	0	0	0	1	0	0	1	1	1	1	112.07	1899
C10	**S14**	0	0	0	0	0	0	1	1	1	1	79.93	917
C11	**S48**	0	0	0	0	0	0	1	1	1	1	102.00	1492

Table 18 Cluster Summary information for RALIC case study

We can see that the clusters are widely distributed along the Pareto front with many areas of high overlaps. One such area is between cost 1400 and 1900 as shown in Figure 49. We use the alternative visualization to identify which clusters are overlapping at those regions. From Figure 48, we can see that in the cost range of 1400 and 1900, we have 7 overlapping clusters, namely C5, C15, C12, C11, C26, C9 and C6.

From the Composition and Distance view in Figure 51, we can see that cluster C14 is the largest cluster with 43 solutions followed by C1 and C21 with 40 and 33 solutions respectively. There are 5 single solution clusters, namely, C10, C3, C17, C5 and C13.

We can also zoom in on the Pareto front view (with the cost against value) to visualise the overlapping clusters. We achieve this by using the zooming facility in the tool to zoom on the clusters between the cost 1400 and 1900. The zoomed view in Figure 49 depicts the overlap with C26 completely overlapping C11, C12 and C15. C9 is the smaller cluster

of the overlapping clusters in that cost region. The alternative visualisation in Figure 50 confirms the overlaps.

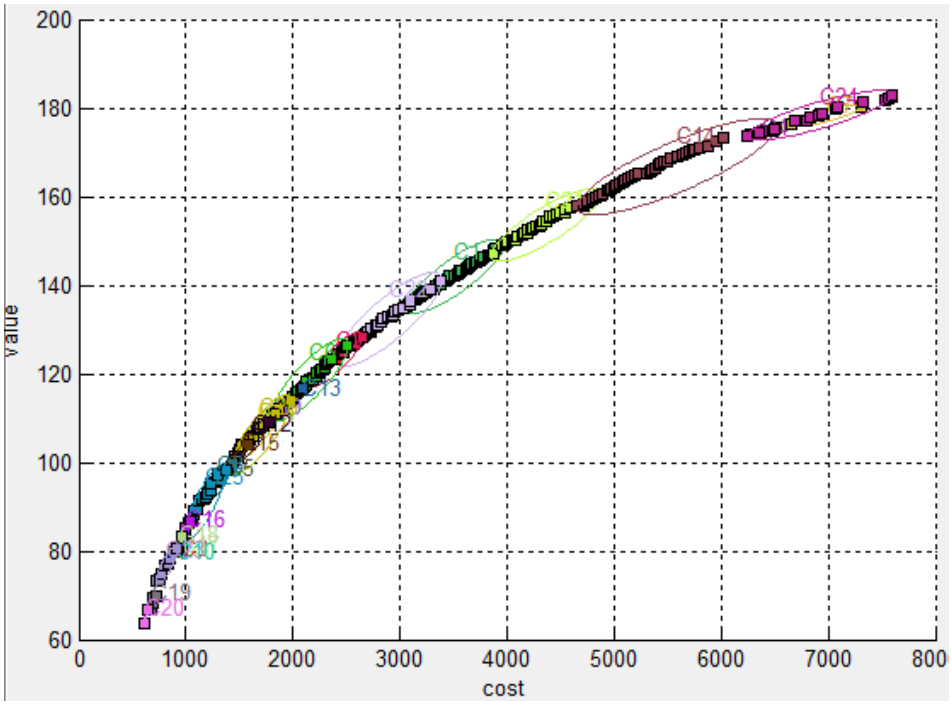


Figure 47 Distribution of RALIC solution clusters on Pareto front

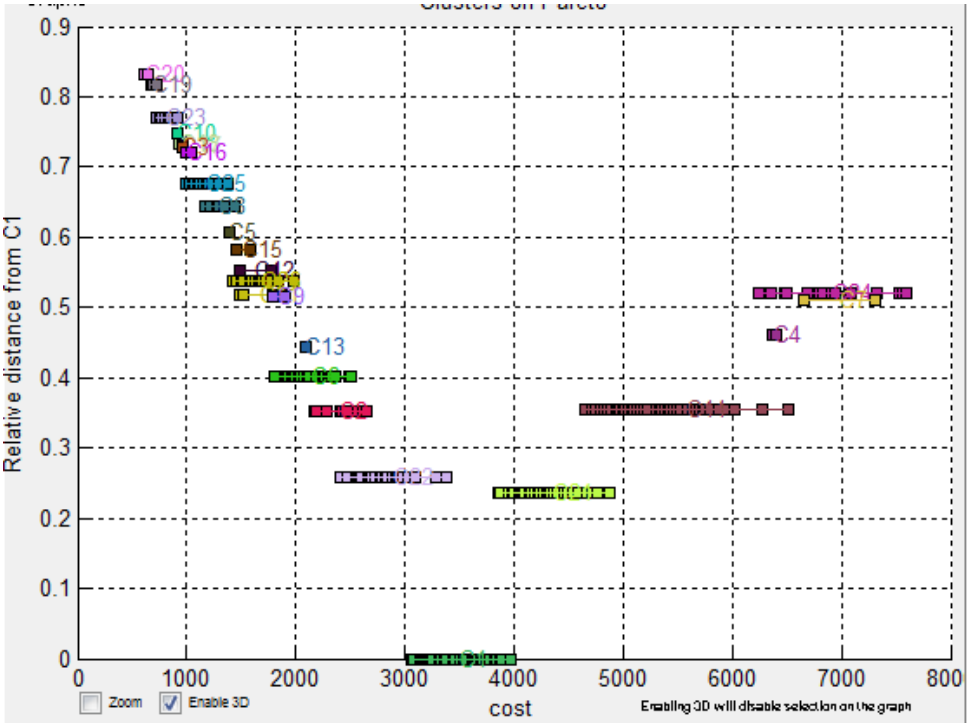


Figure 48 Alternative view of distribution of RALIC Clusters on Pareto front

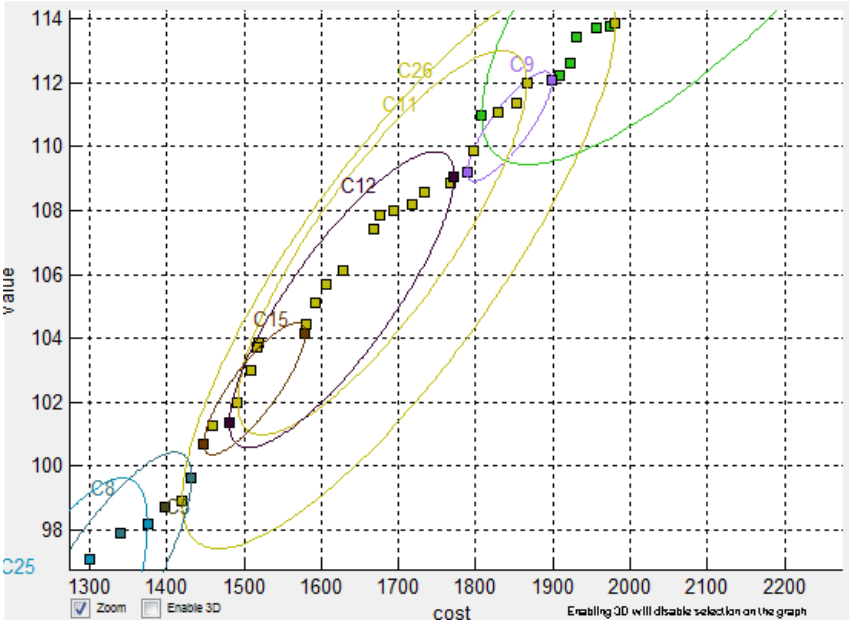


Figure 49 Zooming on region of high overlap between cost 1200 and 1500

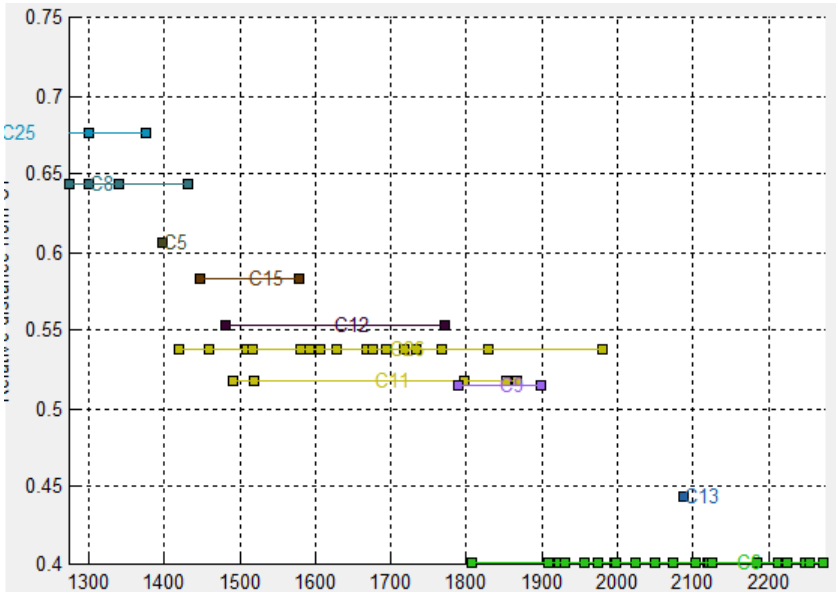


Figure 50 Alternative view for zoomed cost range 1400-1900

To give requirements decision makers a first feel of how the requirements are distributed among the solutions in the clusters, the tool generates the cluster bar chart composition view for the requirements in order of increasing cost as shown in Figure 52. The cluster 'Ph' chart in Figure 53 shows a more detailed composition view with the clusters have been sorted in increasing order of cost. We can see that some requirements such as R67 are never included in the solutions and some requirements such as R27 are included in all the solutions.

	Cluster	Size	Min Cost	Max Cost	Min Value	Max Value	C-Index
1	C20	2	608	643	63.7857	66.7143	0.0348
2	C19	4	668	716	66.8571	69.9286	0.2500
3	C23	8	720	922	73.1429	80.5000	0.6540
4	C10	1	917	917	79.9286	79.9286	0
5	C18	2	932	969	80.6429	83.1429	0.0196
6	C3	1	957	957	80.7143	80.7143	0
7	C17	1	962	962	81.5000	81.5000	0
8	C16	3	990	1043	83.9286	86.7143	0.0675
9	C25	14	996	1375	84.1429	98.2143	0.8930
10	C8	6	1173	1432	91.7143	99.6429	0.3251
11	C5	1	1397	1397	98.7143	98.7143	0
12	C26	16	1420	1981	98.9286	113.8571	1.0778
13	C15	2	1448	1579	100.7143	104.1429	0.0611
14	C12	2	1481	1772	101.3571	109.0714	0.0875
15	C11	5	1492	1866	102.0000	112.0000	0.3470
16	C9	2	1789	1899	109.2143	112.0714	0.0302
17	C6	29	1808	2511	111.0000	126.4286	2.3599
18	C13	1	2087	2087	116.6429	116.6429	0
19	C2	10	2173	2642	118.2857	128.2857	0.7651
20	C22	27	2405	3387	123.5000	141.1429	2.2843
21	C1	40	3057	3964	135.0714	148.9286	2.6668
22	C21	33	3853	4872	146.7857	160.3571	2.4093
23	C14	43	4656	6517	157.7143	175.7143	2.7619
24	C24	17	6243	7600	173.7143	183.0000	1.5882
25	C4	2	6374	6400	174.5000	174.7857	0.0012
26	C7	2	6662	7306	176.2143	180.2857	0.0667

Figure 51 Cluster information for selected clusters

One interesting information that is uncovered is the fact that single-solution cluster C5 is very different from C8 and C26 which are next to it on the charts. We can see that the solution in C5 exclude the requirements R74, R77 and R64 which are present in all solutions in C8 and 26. However, C5 has R36 in which is absent in both C8 and C26.

The descriptions of some of the requirements that vary among these clusters are listed in Table 19. From these descriptions, we can deduce that if we implement solutions in cluster C5, we will lack in security aspects. No secure data storage and no card readers means the data storage will be liable to attacks and there will be no control of who is accessing the university premises and systems that fall under the project.

Identifier	Code	Description	Cost	C8	C5	C26
R11	**b.1.3**	card to include UPI	4	All	None	All
R14	**b.1.7**	card to include expiry date	4	None	All	All
R64	**g.4.5**	ensure secure data storage	79	All	None	All
R74	**i.1.3**	card readers	13	All	None	All
R77	**i.2.5**	documented processes of dependent systems	27	All	None	All
R78	**i.3.1**	requirements management	39	None	All	None

Table 19 Requirements Selections for C8, C5 and C26

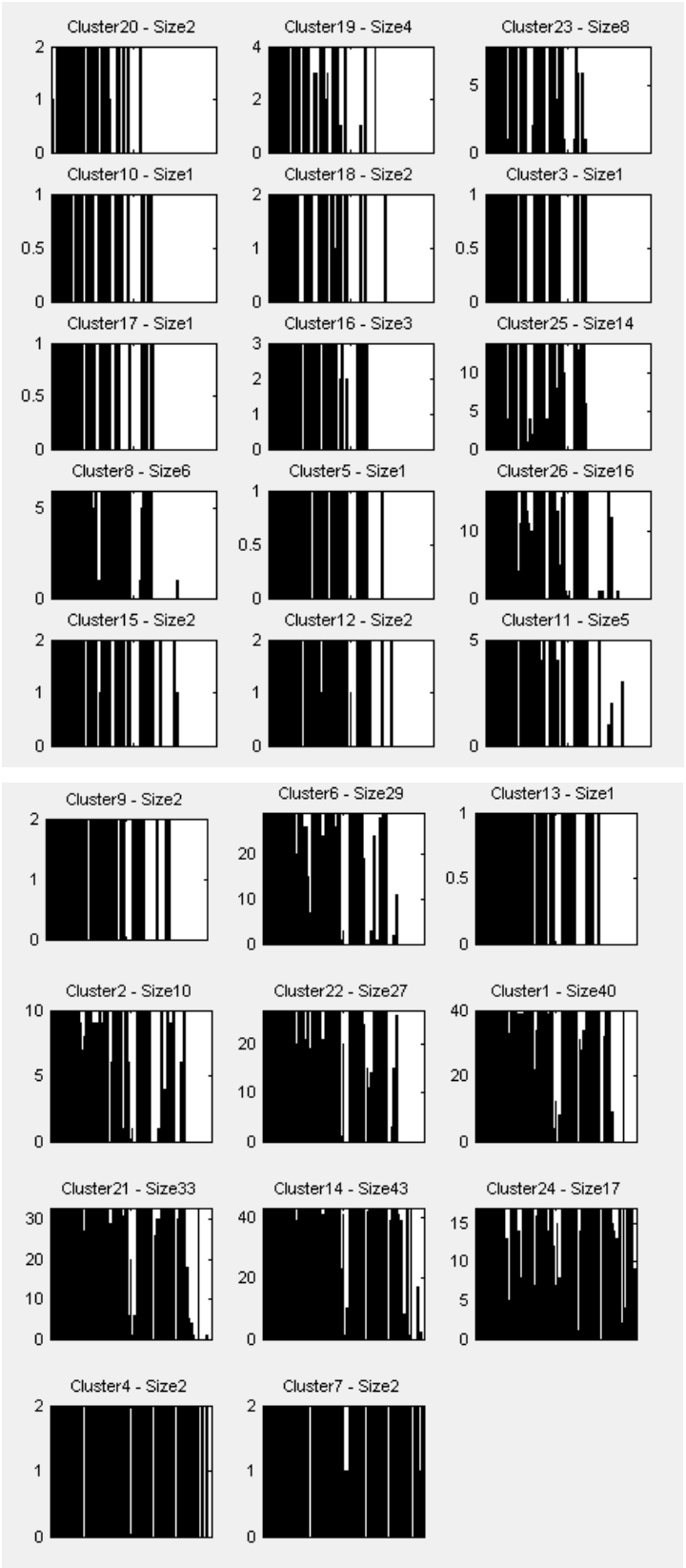


Figure 52 Cluster Composition for RALIC solutions between cost 1200 and 1500

		C20	C19	C23	C10	C18	C3	C17	C16	C25	C8	C5	C26	C15	C12	C11	C9	C6	C13	C2	C22	C1	C21	C14	C24	C4	C7
1	R9	Some	All	All	All	All	All	All	All	All	All	All	Some	All	All	All	All	Some	All	All	All	All	All	All	All	All	All
2	R11	None	All	All	All	All	All	All	All	All	All	None	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
3	R14	None	All	All	All	None	All	All	All	All	None	All	All	All	All	All	All	Some	All	All	All	All	All	All	All	All	All
4	R27	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
5	R33	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
6	R10	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
7	R25	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
8	R26	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
9	R28	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
10	R29	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
11	R34	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
12	R8	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
13	R21	Some	None	Some	None	None	All	All	All	Some	None	All	Some	All	All	All	All	Some	All	All	All	Some	Some	Some	Some	All	None
14	R12	All	None	None	None	All	None	All	All	Some	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
15	R30	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
16	R35	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
17	R20	None	None	None	None	All	None	All	None	Some	All	All	All	All	None	All	All	Some	All	All	Some	All	Some	All	Some	All	All
18	R68	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
19	R74	None	All	All	All	None	All	All	None	All	All	None	All	All	None	All	All	Some	All	Some	Some	Some	Some	Some	Some	All	All
20	R75	All	None	None	None	None	None	All	None	All	All	Some	All	All	None	All	All	All	All	Some	All	All	All	All	Some	All	All
21	R88	None	All	All	All	None	All	All	None	All	All	All	Some	None	None	All	All	Some	All	Some	Some	Some	Some	Some	Some	None	All
22	R93	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
23	R91	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
24	R23	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
25	R80	All	None	All	All	None	None	All	Some	Some	All	Some	All	All	All	All	All	Some	All	All	All	All	All	All	All	All	All
26	R94	None	None	None	None	All	None	All	All	None	None	None	Some	None	All	Some	None	Some	None	Some	Some	Some	Some	Some	Some	All	All
27	R13	All	None	None	None	None	None	None	None	Some	All	None	Some	All	Some	All	None	Some	All	Some	Some	Some	All	All	Some	All	All
28	R81	All	Some	None	None	None	None	None	All	None	Some	All	Some	None	All	All	All	Some	All	Some	All	Some	All	Some	All	All	All
29	R90	None	Some	Some	All	None	None	None	None	Some	Some	None	Some	None	None	None	None	Some	All	Some	Some	Some	Some	Some	Some	Some	None
30	R65	None	None	All	All	None	All	All	None	All	All	All	Some	All	All	All	All	Some	All	Some	Some	Some	All	All	All	All	None
31	R72	All	None	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
32	R76	All	All	All	All	All	All	All	None	All	Some	All	All	All	Some	All	All	Some	All	Some	All	All	All	All	All	All	All
33	R77	None	None	None	None	All	None	All	None	None	All	None	All	None	Some	All	All	All	Some	Some	Some	All	All	Some	All	All	All
34	R86	All	All	All	All	All	All	All	All	All	All	All	All	All	All	Some	All	All	All	All	All	All	All	All	All	All	All
35	R87	All	Some	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
36	R63	Some	Some	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
37	R18	None	None	None	None	None	None	None	All	Some	All	None	None	None	All	None	All	Some	None	None	Some	Some	Some	Some	Some	All	All
38	R16	None	None	None	None	All	None	None	None	Some	Some	All	None	None	None	None	All	Some	All	Some	Some	Some	Some	Some	Some	All	All
39	R49	None	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
40	R61	All	None	None	None	None	None	None	None	None	Some	None	None	None	None	Some	Some	None	Some	Some	Some	Some	Some	Some	None	None	None
41	R54	All	All	All	All	Some	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
42	R40	None	All	All	All	All	All	None	None	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
43	R78	None	None	None	All	None	None	None	None	Some	None	All	None	All	None	None	All	Some	All	Some	Some	Some	Some	Some	All	All	All
44	R97	All	Some	Some	None	All	All	None	Some	Some	All	All	Some	All	All	Some	All	All	All	All	All	All	All	All	All	All	All
45	R85	None	None	All	None	None	None	None	All	All	All	All	Some	None	All	None	None	Some	None	Some	Some	All	Some	Some	All	All	All
46	R22	None	None	None	None	None	None	None	None	Some	None	None	None	None	None	None	None	None	None	None	None	Some	Some	Some	Some	None	None
47	R32	All	All	All	All	All	None	None	None	All	All	All	Some	All	All	All	All	All	All	All	All	All	All	All	All	All	All
48	R24	None	None	Some	None	All	None	All	Some	Some	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
49	R89	None	None	None	None	None	None	None	None	None	None	None	Some	None	None	None	All	Some	All	Some	Some	Some	Some	Some	Some	All	All
50	R79	None	None	None	None	None	None	None	None	None	None	None	None	Some	None	None	Some	None	Some	None	Some	Some	Some	Some	Some	None	All
51	R98	None	None	None	None	None	None	None	None	None	None	None	Some	None	None	None	None	None	None	Some	None	None	Some	Some	Some	All	Some
52	R99	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	Some	Some	Some	All	Some
53	R73	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	Some	Some	None	Some
54	R45	All	None	Some	None	None	All	None	All	All	Some	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
55	R66	None	None	All	All	None	All	All	None	All	Some	All	Some	None	Some	All	Some	Some	All	All	All	All	All	All	All	All	All
56	R7	None	Some	Some	All	All	All	All	All	Some	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
57	R17	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	Some	Some	Some	All	Some
58	R44	None	None	None	None	None	None	None	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
59	R42	None	All	Some	All	All	All	None	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
60	R41	None	None	None	None	None	None	None	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All	All
61	R64	None	None	Some	All	None	All	All	None	Some	All	None	All	None	Some	All	All	All	All	All	All	All	All	All	All	All	All
62	R38	None	None	None	None	None	None	All	None	None	None	None	All	All	All	All	None	All	All	All	All	All	All	All	All	All	All
63	R15	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	None	None	Some	All	All	All	All	All	All
64	R60	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	None	None
65	R69	None	All	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	Some	Some	Some	All	All
66	R71	None	None	None	None	None	None	None	None	None	None	None	None	All	None	None	None	None	None	None	Some	Some	Some	All	All	All	All
67	R46	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	None	Some	Some	Some	Some	All	All	All	All
68	R55	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	None	None
69	R36	None	None	None	None	None	None	None	None	None	None	All	Some	None	All	All	All	Some	All	All	All	All	All	All	All	All	All
70	R62	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None
71	R83	None	None	None	None	All	None	None	None	None	None	None	Some	None	None	None	None	Some	All	Some	All	All	All	All	All	All	All
72	R51	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	Some	None	All	All	All	All	All
73	R2	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	All	All	All	All	All	All	All	All	All
74	R48	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	Some	All	All	All	All	All
75	R43	None	None	None	None	None	None	None	None	None	None	None	All	All	All	Some	All	All	None	Some	All	All	All	All	All	All	All
76	R82	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	None	None
77	R4	None	None	None	None	None	None	None	None	None	Some	None	Some	Some	None	Some	All	All	All	All	All	All	All	All	All	All	All
78	R67	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None
79	R53	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	Some	All	All	All
80	R39	None	None	None	None	None	None	None	None	None	None	None	Some	None	None	None	None	None	None	None	Some	Some	All	All	All	All	All
81	R1	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	None	Some	Some	All	All	All	All	All	All
82	R50	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	All	All	All	All
83	R6	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	None	Some	None	All	Some	All	All	All	All	All	All	All
84	R57	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	All	None	All
85	R31	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	Some	Some	Some	All	All
86	R52	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	Some	Some	None	All
87	R19	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	Some	All	All
88	R47	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	Some	Some	All	None
89	R84	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	Some	Some	All	All
90	R58	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	All	All	All	None
91	R92	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	Some	All	All
92	R5	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	All	All	All	All	All
93	R70	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	None	None
94	R56	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	All	All	All
95	R95	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some
96	R37	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	All	None	All
97	R96	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	All	All	Some
98	R59	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	Some	None	None
99	R3	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	None	Some	None	All

Figure 53 Cluster "Ph" Chart for RALIC solutions

Another interesting observation can be made about clusters C19 and C20.There is a discontinuity in the requirements as cost and value increases from C19 to C20. The pair-wise comparison of these clusters is shown in Figure 52. We find that C19 has R11, R14, R40, R42, R49, R69, R74 and R88 in all of its solutions which are absent in all of the solutions in C20. This shows that it is wrong to always assume that a higher cost cluster of solutions will be a superset of a lower cost one in terms of design.

		C20		
		None	Some	All
C19	None	R1 R2 R3 R4 R5 R6 R15 R16 R17 R18 R19 R20 R2...	R21	R12 R13 R45 R61 R72 R75 R80
	Some	R7 R90	R63	R81 R87 R97
	All	R11 R14 R40 R42 R49 R69 R74 R88	R9	R8 R10 R23 R25 R26 R27 R28 R29 R30 R32 R33 R34 R35 R54 R68 R76 R86 R91 R93

Figure 54 Pair-wise cluster comparison for C19 and C20

The description of some of the differing requirements in C19 and C20 are listed in Table 20. We can further investigate how the clusters are similar or different from this information. For example, solutions in C19 do not provide maintenance facilities such as remote updates and software upgrades for the system which are present in solutions in C20. Solutions in C20 on the other hand restrict the options to manage the access cards. Thus, it does not cater for expiry dates on the cards, their activation or handing of lost cards.

Identifier	Code	Description	Cost	C19	C20
R11	**b.1.3**	card to include UPI	4	All	None
R12	**b.1.4**	card design to include card type/user status	11	None	All
R13	**b.1.5**	card to include payroll number	24	None	All
R14	**b.1.7**	card to include expiry date	4	All	None
R40	**d.4.1**	ucl shop to handle lost cards	38	All	None
R42	**d.5.2**	activate and inactivate card	77	All	None
R45	**d.6**	able to create access reports	75	None	All
R49	**f.1**	compatible with Bloomsbury system (Gladstone MRM)	35	All	None
R61	**g.4.2**	update/delete data remotely	35	None	All
R72	**h.4**	upgradable (software revisions)	27	None	All

Table 20 Requirements Selections for C19 and C20

If the decision maker wants to confirm the overlaps with a cost/value range or if the decision maker already knows that he wants to look only at specific cost/value on the Pareto front, he can generate the clusters only for this range.

We next run the clustering in the range 1400 to 1900 as we want to confirm if the area is really one of great overlap and whether the single solution cluster is really a single solution one. We choose a cut-off that will generate 7 clusters. The resulting distribution on the Pareto front is illustrated in Figure 55. This confirms the high area of overlapping.

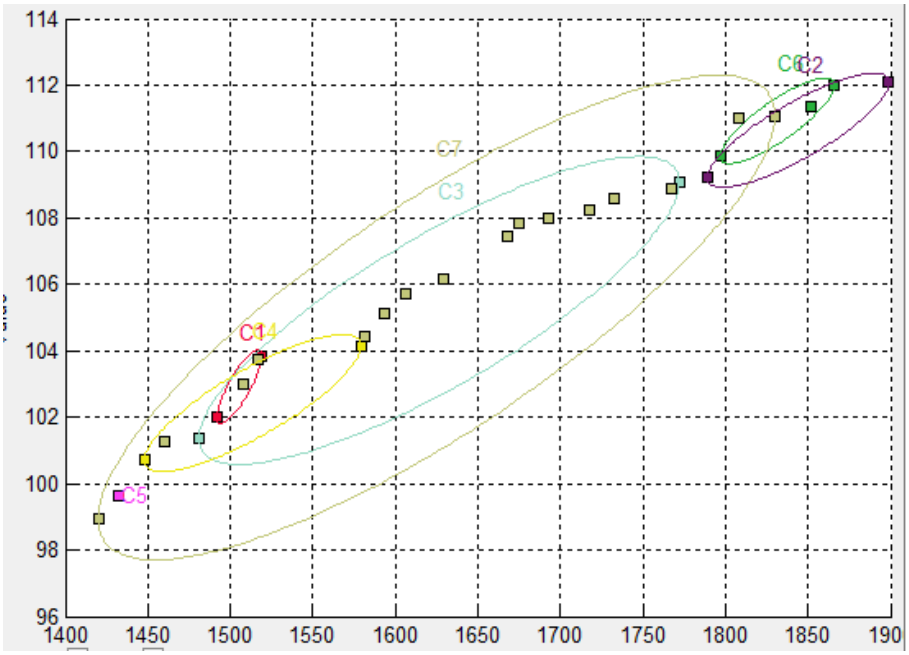


Figure 55 Clusters for cost range 1400 to 1900

Conclusion

Starting from 274 solutions, our technique has reduced that number to 26 groups of solutions from which the requirements decision maker can make an initial choice. We have been able to see the areas on the Pareto front where we have differing solutions within same cost range. We have been able to identify variations in the design of these solutions - for example a low cost cluster C18 having a high cost requirement R83 in both of the solutions in it. Our cluster composition views also show that the 12 lowest cost requirements tend to be included in all the solutions discovered by the NSGA-II technique for this dataset. However, there are exceptions when one or more of these requirements are omitted (for example C28, C18, C8 and C5). Our technique has enabled us to identify these exceptions easily.

6.5 Clustering Pareto Optimal Solutions for Motorola

We will next cluster the Pareto optimal solutions discovered by the NSGA-II search-based optimisation technique (without dependencies) on the Motorola case study using the *optimal solutions analysis technique*. We first feed the data elicited from the 4 stakeholders for the 35 requirements identified by Motorola. 108 solutions were discovered by the search-based technique in this case with cost ranging from 20 to 6740 and value ranging from 1.5 to 19.5. The resulting Pareto front is depicted in Figure 56. The Pareto front consists of isolated solutions at the beginning (around cost 0) and at the end (cost 4000 onwards) rather than being a continuous one.

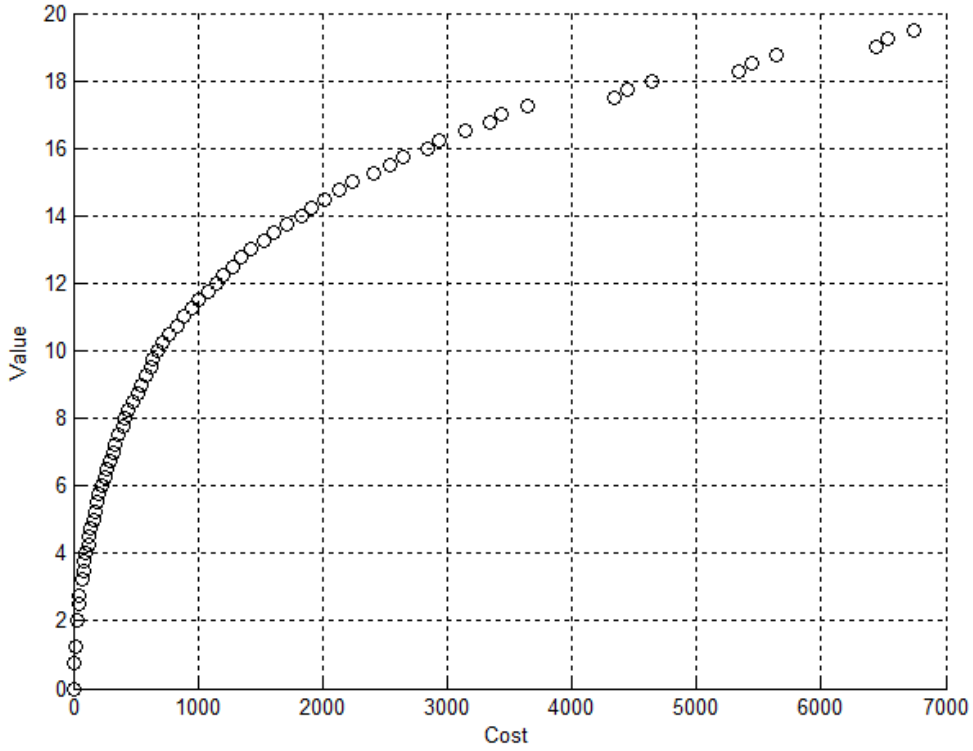


Figure 56 Pareto front for Motorola dataset

We use this set of optimal solutions as input into our technique to generate the dendrogram in Figure 57. We have weighed the distance measure for the clustering process by cost. The recommended default cut-off is shown with a black dotted line. The default cut-off generates 10 clusters and this cut-off value is a good one from the value of the quality indices (on Figure 58) for the clusters generated as it gives a good balance between these indices.

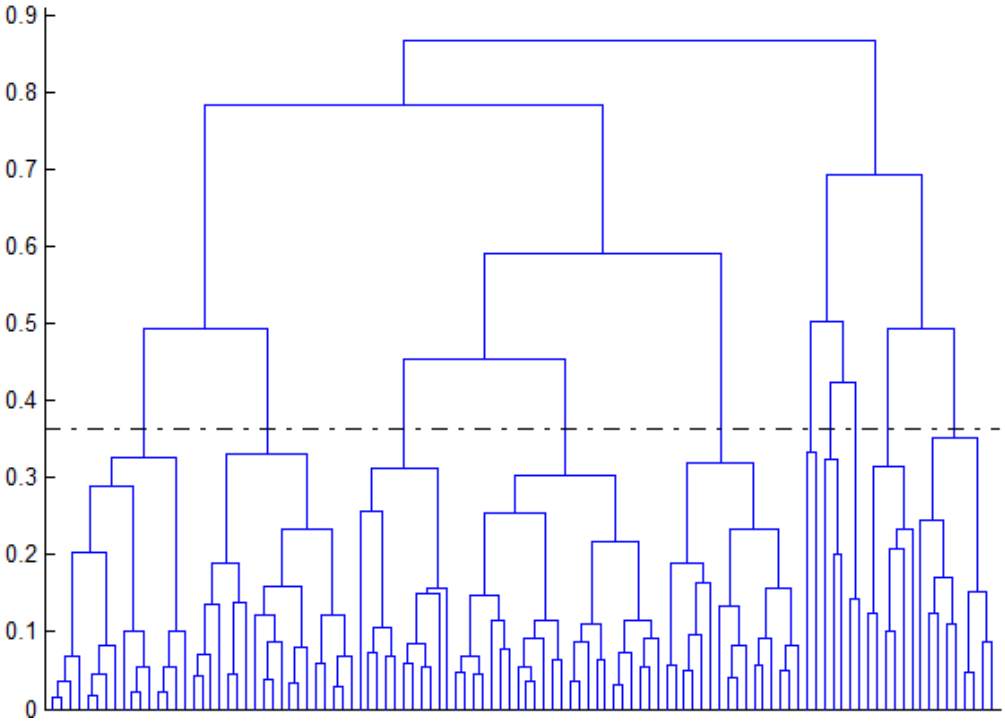


Figure 57 Dendrogram for Motorola dataset

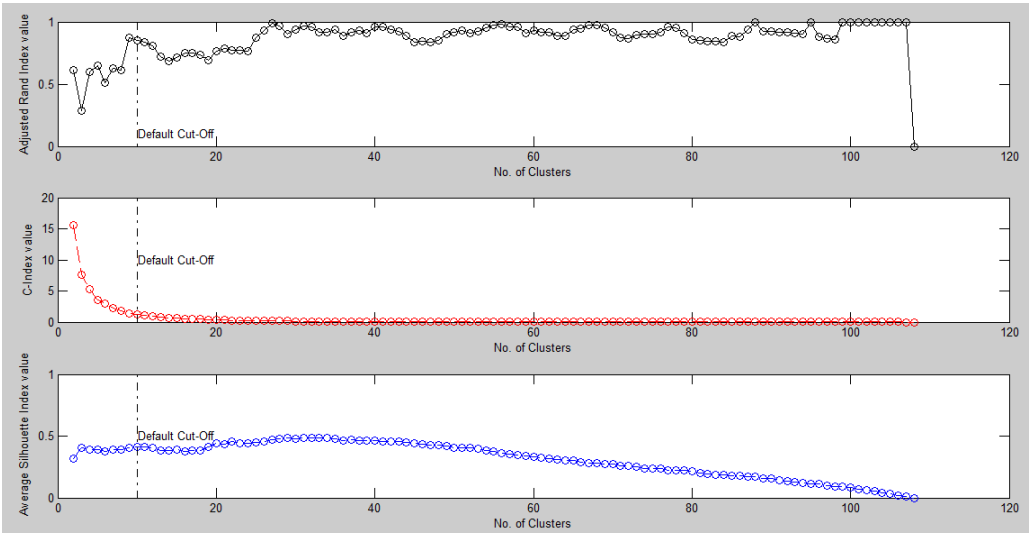


Figure 58 Rand and C-Index values for Motorola dataset

Our technique first identifies how the clusters are distributed on the Pareto front using different colours for solutions according to the clusters in which they fall and coloured ellipses to show the range of the clusters. The distribution of clusters is shown

in Figure 59. Figure 60 zooms in on the area of the Pareto front where some overlap occurs.

We can see that the isolated solutions at the beginning of the Pareto front are indeed very different from each other while those towards the end of the front, despite being isolated from each other, are in the same cluster and hence, similar. We can also see that we have 3 clearly delimited clusters C6, C10 and C4 with no overlap at all at the end of the Pareto front. The alternative view in Figure 61 enables us to confirm that we have slightly overlapping smaller clusters in the beginning of the front. The area from cost 30 to cost 230 has some overlaps between clusters C7, C1, C3, C8 and C9.

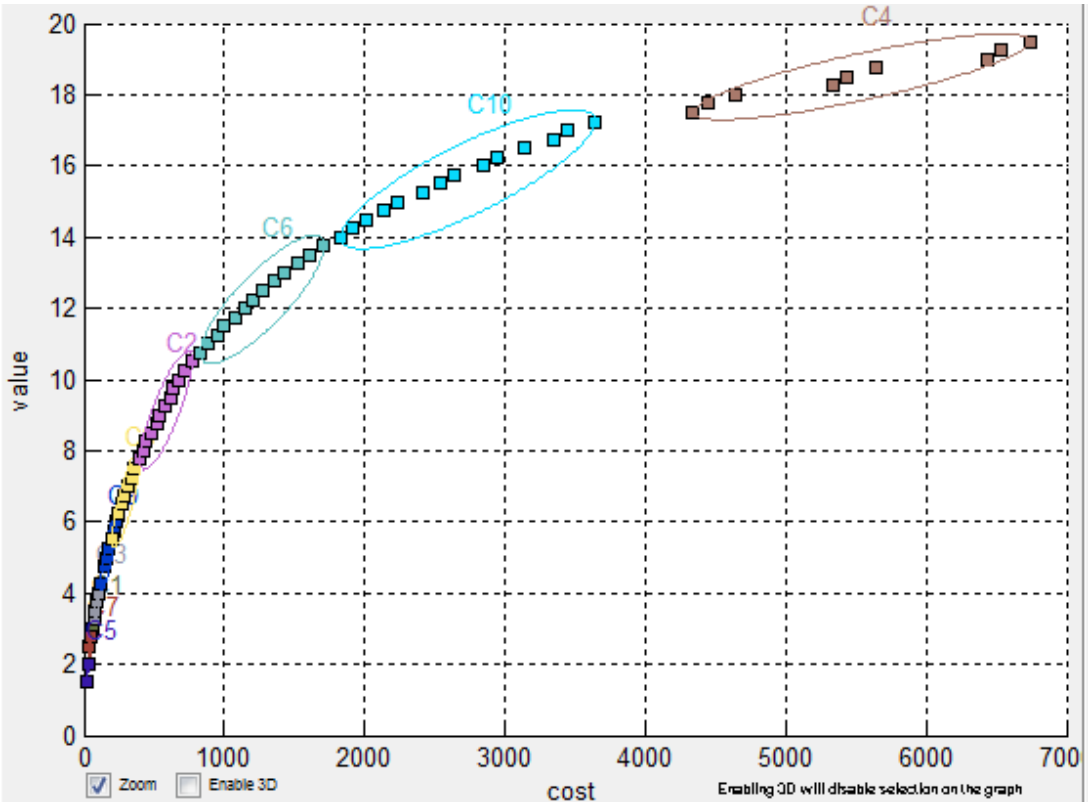


Figure 59 Distribution of Motorola solution clusters on Pareto front

As we can see from Figure 62 which gives the cluster information for the clusters, we have six clusters with 11 or less solutions and one big cluster, C2 with 24 solutions.

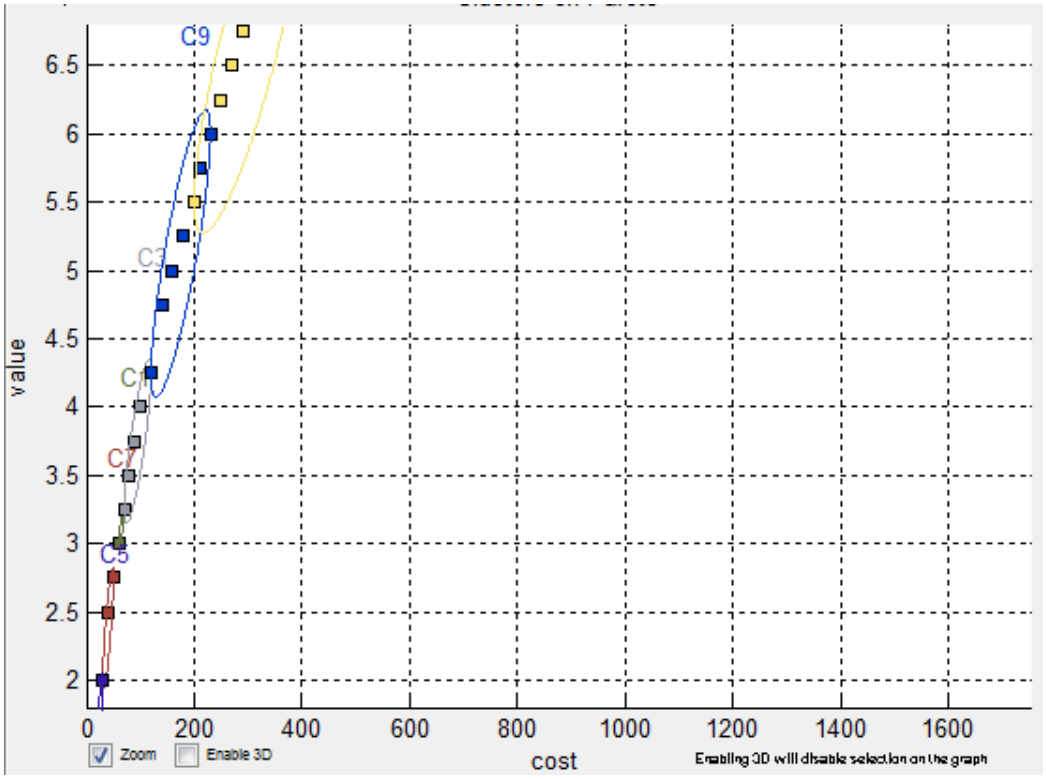


Figure 60 Zooming on region of overlap between cost 100 and 300

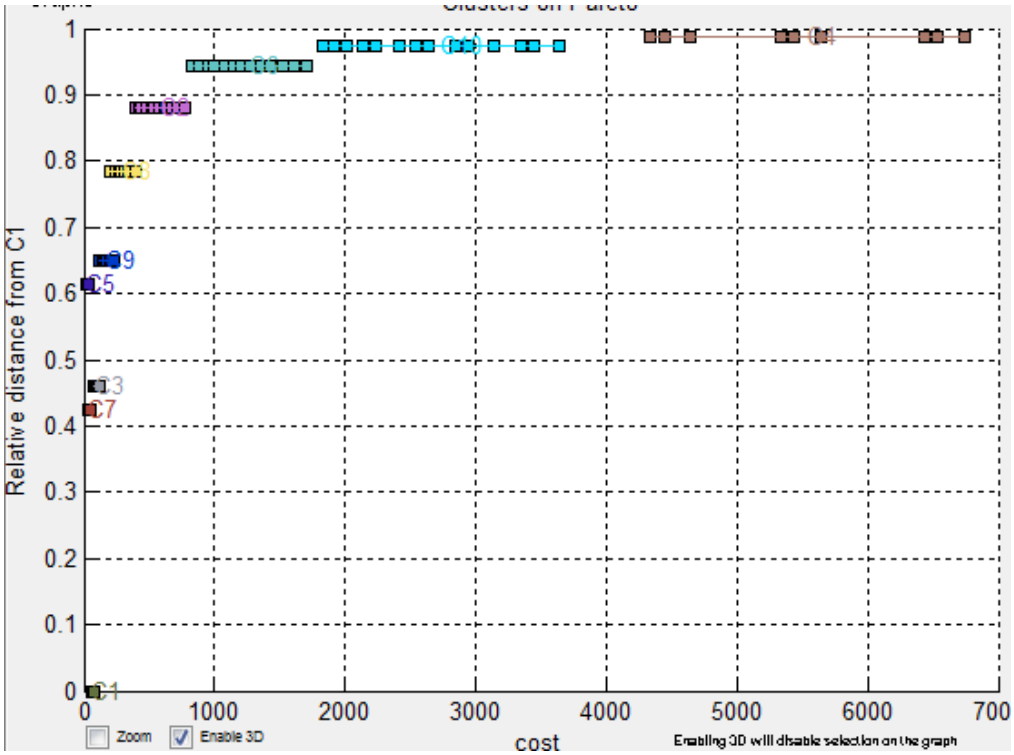


Figure 61 Alternative view of distribution of Motorola Clusters on Pareto front

	Cluster	Size	Min Cost	Max Cost	Min Value	Max Value	C-Index
1	C5	2	20	30	1.5000	2	0.1607
2	C7	3	30	50	2	2.7500	0.2805
3	C1	2	60	70	3	3.2500	0.0645
4	C3	6	70	120	3.2500	4.2500	0.6456
5	C9	9	120	230	4.2500	6	1.0759
6	C8	11	200	400	5.5000	7.7500	1.1656
7	C2	24	400	770	7.7500	10.5000	2.7972
8	C6	16	830	1710	10.7500	13.7500	1.8489
9	C10	19	1830	3640	14	17.2500	2.2620
10	C4	16	4340	6740	17.5000	19.5000	1.8621

Figure 62 Cluster/Distance information for Motorola Solutions Clusters

Our technique also generates the cluster summary table for the clusters listing the individual solutions and the cluster to which they belong. Part of this table for the first 10 requirements in shown in Table 21.

Cluster	Solution	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	Value	Cost
C1	**S6**	1	0	1	1	1	0	0	1	0	0	3	60
C1	**S7**	1	1	1	1	1	0	0	1	0	0	3.25	70
C2	**S34**	1	1	1	1	1	0	0	1	1	1	7.75	400
C2	**S35**	1	1	1	1	1	0	1	1	1	1	8	420
C2	**S36**	1	1	1	1	1	1	0	1	1	1	8	420
C2	**S37**	1	1	1	1	1	1	1	1	1	1	8.25	440
C2	**S38**	1	1	1	1	1	1	1	1	1	1	8.5	480
C2	**S39**	1	1	1	1	1	1	1	1	1	1	8.5	480
C2	**S40**	1	1	1	1	1	1	1	1	1	1	8.75	520
C2	**S41**	1	1	1	1	1	0	1	1	1	1	8.75	520

Table 21 Cluster Details Table for Motorola Case Study

If we look at the cluster composition bar chart view in Figure 63 and the cluster "Ph" chart in Figure 64, we can observe that the number of requirements being included increases as the overall cost of the clusters increases. The cheap requirements R1 and R3 are present in all solutions along the Pareto front. However, the most expensive requirement R35 is added very late on the Pareto front and that in only some of the solutions.

R9 is present in all solutions only as from C3, R10 is present in all solutions as from C9 while R15 is present in all solutions in from C8. R17 is present in all solutions from C2 onwards. We can also observe that the last three clusters C6, C10 and C4 incrementally add R18 until all solutions in C4 include it.

It may also be of interest to observe how the presence of R8 in the solutions vary with the overall composition of C1, C3 and C9 as it is being removed while other requirements are being added in some of the solutions indicating the kind of trade-off being done in the design.

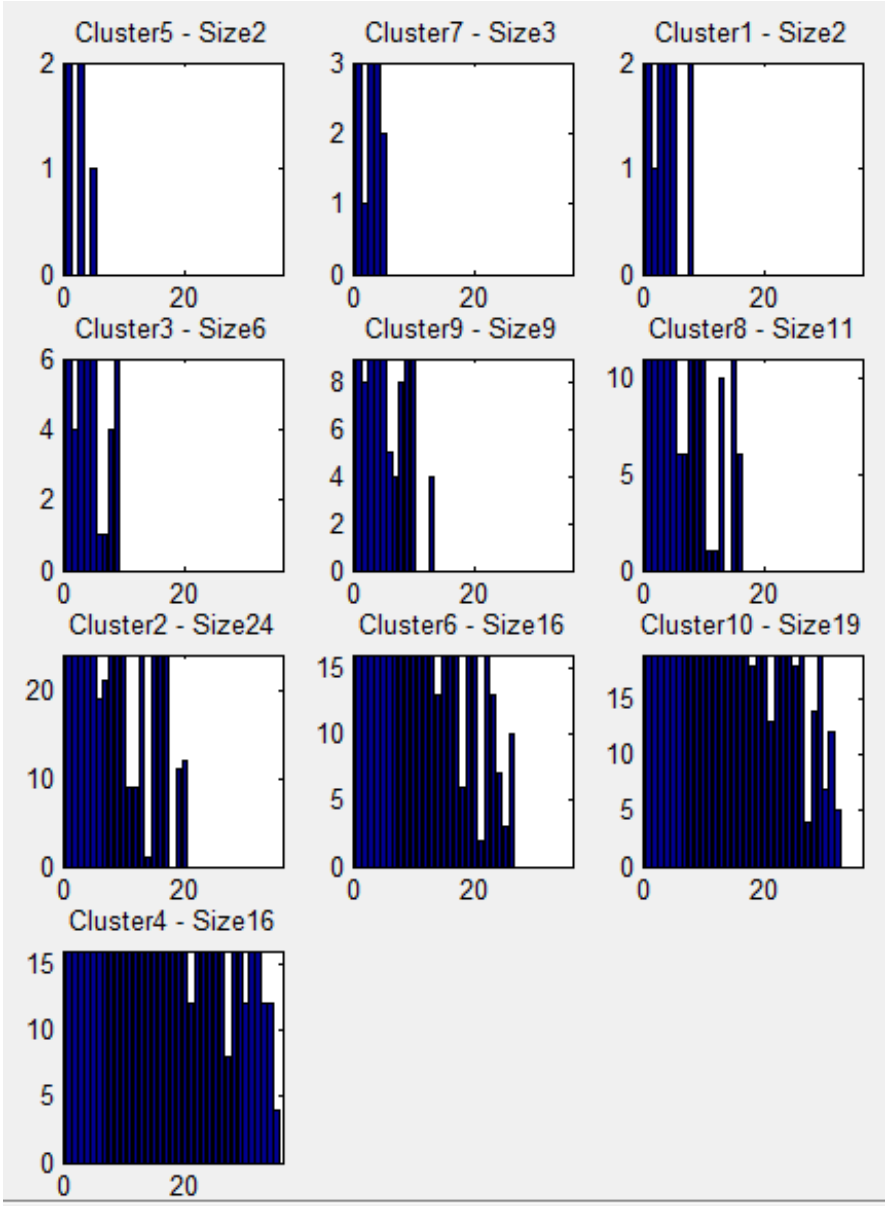


Figure 63 Cluster Fingerprint for Motorola Solutions Clusters

Figure 64 Cluster "Ph" Chart for Motorola solutions

One significant overlap on the Pareto front occurs between C8 and C9. The requirements decision-maker may want to investigate these clusters alone using the pair-wise comparison screen on our tool. The result of the comparison is shown in Figure 65 where we can see that R15 is present only in C8, R2 and R8 are present only in some solutions in C8 only while R1, R3, R4, R5, R9 and R10 are present in both clusters.

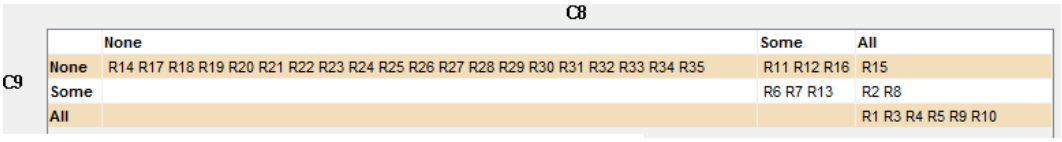


Figure 65 Pair-wise comparison for C8 and C9

This shows that as the cost of the clusters increases from C9 to C8, requirements have only been added to the solutions.

Conclusion

We have produced 10 clusters of solutions from the initial 108 solutions on the Pareto front which is a significantly lower number of initial groups to choose from. Clustering the solutions for the Motorola dataset has enabled us to see that the solutions tend to highly differ from each other with only slight cluster overlaps and many clearly separated clusters. The *optimal solutions analysis* technique has found a large number of small clusters early on the Pareto front. However, as the cost increases, more distinct, larger clusters are formed. We have also been able to see how the requirements are being incrementally added to the solutions as the cost increases.

6.6 Threats to Validity

6.6.1 Internal Validity

Dependencies and conflicts always exist among requirements. During the *stakeholders' preferences analysis*, we do not include these in the heuristics of the clustering as we want to group stakeholders according to their preferences independently from potential conflicts and dependencies between requirements. The *optimal solutions analysis* technique assumes that all the dependencies and conflicts among requirements have been handled by the search-based technique that has been used to generate the Pareto optimal front.

Our *stakeholders' preferences analysis* technique enables us to infer preferences for the different clusters we have discovered. Since the data we have used for our validation is one from a past project, we have not been able to investigate with the stakeholders and confirm that the preferences we found actually reflected their actual preferences. However, this extra investigation exercise could provide further information on how to improve the elicitation process as it can help clarify any misunderstandings that the stakeholders might have about the requirements.

The clusters of optimal solutions discovered by our *optimal solutions analysis* technique can be overwhelming when we have many overlaps. This happens in systems where only small changes in the combinations of requirements can produce highly different solutions in terms of the objectives being measured. This is independent of the requirements' dependencies and conflicts that have already been considered in the search-based prioritisation technique that has found the Pareto optimal solutions. We have tried to address this with a "Zoom" option on the tool that enables the decision maker to only view the clusters on a specific area of the Pareto front.

We have provided a non-exhaustive list of views and analyses that can be done on the outcome of both our techniques. These views and analyses were most relevant on the case studies and randomly generated data that we used. The users of our techniques can easily use the exported data from the tool to feed into other statistical analysis / visualisation tools to perform more in-depth analysis that they may need.

6.6.2 External Validity

We have designed the *stakeholders' preferences analysis* technique for large numbers of stakeholders in the context of web-based requirements elicitation tool. We have been able to demonstrate the usefulness on our technique on the RALIC dataset where we had only 76 stakeholders with 99 requirements. We believe that our technique can be applied successfully in the target contexts although the size of the RALIC dataset is not as large as the ones expected in these contexts.

The *optimal solutions analysis* technique aims at providing insights on generated Pareto fronts. The case studies we have looked had 274 and 108 solutions with quite different Pareto fronts. In each case, we have been able to get useful insights on the solutions. This clearly demonstrates that our optimal analysis technique can be useful to understand any type of Pareto fronts where design trade-offs need to be understood.

The evaluation of the cost effectiveness of the stakeholders' preference analysis with regards to the overall software development life cycle is more subjective. For example, if the outcome of the stakeholders' preferences analysis indicates the need for further elicitation, this will increase the cost of requirements engineering but may yield a result in a low cost of rework in the later phases. The *optimal solutions analysis* technique however helps to decrease the effort required to understand the design trade-offs on the Pareto front and aids towards the choice of the best possible alternative. This should result in a gain in cost over the overall project.

6.7 Conclusions from Case Studies

The first objective of the case studies we have looked at was to check whether our *stakeholders' preference analysis* allows us to form stakeholder clusters that have preference values closer to the actual individual preferences of the stakeholders than other approaches. We also verify if our approach allows us to identify trends in the preferences of the stakeholders as well as 'outliers' stakeholders in the case studies.

The second objective was to check whether our *optimal solutions analysis* helps us to identify and understand variations within the set of optimal solutions by grouping them according to their design similarities.

This section discusses the extent to which our techniques enable us to meet these objectives and the lessons we have learnt by applying them.

6.7.1 Stakeholders' Preference Analysis

We have found that grouping stakeholders by preference helps to better represent them in the decision than using an overall preference or a stakeholder group preference. In depth analysis of stakeholders' preferences has enabled us to find trends and differences in the preferences of the stakeholders. We have been able to find which stakeholders are "outliers" in terms of preference. This information can be used to further investigate and uncover underlying requirements that might have been overlooked.

Using the *stakeholders' preferences analysis* has shown us that the assumption that all stakeholders within one stakeholder group have similar preferences does not always hold true. For example, in the RALIC case study, we have been able to see how diverging preferences are present in the "admin" stakeholders' group. Another benefit that may arise from our *stakeholders' preference analysis* technique is the handling of missing data. If in a stakeholder cluster we have some stakeholders that have not rated for a particular requirement, our technique automatically infers a preference value for that stakeholder based on the value given to that requirement by other stakeholders with similar preferences.

When there is a large number of stakeholders our *stakeholders' preferences analysis* technique makes the detection of outlier stakeholders very easy. Noticeable improvement in this exercise was achieved in the RALIC case study where there are only 76 stakeholders. In projects with a larger number of stakeholders, this improvement should be even more considerable. Also, clustering the stakeholders by preference resulted in fewer preferences groups to look at. This gave a very good overall idea of the combinations of preferences that were more common among the stakeholders.

However, even after clustering stakeholders by their preference, the spread of the preferences for some requirements in a given stakeholder cluster may still be large. This is the case for requirement *h.2.2* for cluster 9 in the RALIC dataset. This arises because the clustering algorithms has found a high similarity on the preferences for other requirements for the stakeholders in that cluster and grouped them together. This can be improved by using a lower cut-off on the dendrogram to produce smaller clusters. The drawback of this operation is that the number of clusters may increase dramatically and we may end up with a large number of single-stakeholder clusters.

6.7.2 Optimal Solutions Analysis

Grouping solutions by design similarity enables us to see how the solutions vary along the Pareto front in terms of design solutions. Applying the *optimal solutions analysis* technique on the case studies has shown that solutions do not always incrementally add new requirements when the cost increases. We have been able to identify interesting information such as low cost clusters that include expensive requirements. We have validated our technique using both the RALIC and the Motorola datasets in this thesis. In previous work (Veerappa and Letier 2011), we have done a further validation on the dataset used in (Greer & Ruhe, 2004). Our technique has consistently helped us find the differences and similarities among the solutions on the Pareto front in all three cases.

In this thesis, we have found that the RALIC case study produced highly overlapping clusters of solution on the Pareto front. These are indicators of discontinuities that may exist in a given range of cost/value. Clusters 19 and 20 are such an example. Unlike the RALIC solutions, the Motorola solutions did not include significant design discontinuities. In fact, we had clearly separated clusters in some areas of the Pareto front. The only few exceptions are observed early on the Pareto front where we can see that there are only one or two overlapping solutions from each clusters. Our technique therefore allows decision makers to identify when discontinuities are present or not in a set of solutions.

In cases where we have considerable design discontinuities, broadening the Pareto optimal front into a Pareto optimal "corridor" where the fitness functions allow for a range of candidate Pareto solutions could enable our technique to provide more insights in the trade-offs that can be made within that range.

We claim clustering the solutions by their design similarities during the *optimal solutions analysis* reduces the cognitive load as we only have to focus on specific sets of solutions first. The subjective nature of the solutions analysis exercise has made it difficult for us to assess to what extent our technique reduces the cognitive load on decision makers when they are analysing the Pareto front. Even if we try to investigate this aspect of our technique under controlled conditions, it will be very difficult to compare the results.

For both techniques, these observations have only been made on the case studies we have looked at, and further validation on larger data sets must be carried out to evaluate how our techniques behave in those cases.

6.8 Scalability and Performance

The Matlab stopwatch timer is used to determine the amount of time the clustering algorithm takes to execute both in the case of the *stakeholders' preferences analysis* and the *optimal solutions analysis* techniques. We use the 3 cases - 15 stakeholders and 40 requirements, 50 stakeholders and 80 requirements and 100 stakeholders and 140 requirements - used by Zhang (Zhang 2010) and benchmark against the time taken by NSGA-II to find optimal solutions for the each of them.

We proceed by first generating the Pareto front and record the time taken by Matlab in each of the cases. We then run the *stakeholders' preferences analysis* and the *optimal solutions analysis* on the datasets and record the time taken in each case. The results are listed in Table 22. The machine that we used for the tests has an Intel Core i5 CPU and 4GB RAM.

No. Of Stakeholders	No. of Requirements	Time to generate Pareto Front (s)	Number of Pareto Optimal Solutions	Time to perform Stakeholder Preferences Analysis (s)	Time to perform Optimal Solutions Analysis (s)
15	40	105.41	336	2.55	25.89
50	80	129.78	418	2.61	40.64
100	140	183.48	286	12.88	19.51

Table 22 Load test results

The time increases as the size of the data being clustered increases. This confirms the time complexity of hierarchical clustering algorithms which is $O(n^2)$ as these need to compare all possible pairs of objects. However, this remains considerably less than the time taken to generate the Pareto Optimal front itself and can be an acceptable overhead to improve the decision making process.

6.9 Summary

We have applied our *stakeholders' preferences analysis* and *optimal solutions analysis* techniques to the RALIC and Motorola case studies. We have demonstrated how our techniques help to improve the decision making process and discussed what we have learned from the case studies.

Chapter 7 - Conclusion and Future Work

We conclude this thesis and lay out future work in this chapter

7.1 Contributions

Decision making is a key activity in requirements engineering. Making the right decision is determined by two main factors: how well we represent what the stakeholders want in the decision and how well we can interpret the output of the decision making techniques. We have seen these are often hindered in large scale requirements elicitation contexts where we have large numbers of stakeholders and requirements to work from.

7.1.1 Stakeholders' Preferences Analysis

The first contribution of this thesis aims to improve the representation of stakeholders' preferences in the requirements decision-making process. We have seen that when we have a large number of stakeholders, their voices were represented by computing a representative value for either the whole population of stakeholders or the product-independent stakeholder groups that have been elicited during the stakeholder analysis phases. There may be large variations among the stakeholders' preferences within each stakeholder groups. Such differences are usually not visible to the decision maker and may be lost in the decision making process.

Our *stakeholders' preferences analysis* technique reduces this difference between the representative value and the actual stakeholders' preference by first grouping similar stakeholders according to their preferences. We use the hierarchical agglomerative clustering algorithms to form clusters of stakeholders based on their preferences. We then compute the representative value for these clusters of stakeholders that we feed into the decision making techniques.

We have seen that after performing *stakeholders' preferences analysis* on the stakeholders, we get a representative value that is much closer to the preference of the

stakeholders. This contributes towards decisions more likely to reflect the preference of the stakeholders.

The clusters of stakeholders formed from the *stakeholders' preferences analysis* have also provided us with insights about how the preferences of stakeholders differ. We have seen for example that is not true to assume that all stakeholders with the same role in the project will have similar preferences. Another useful outcome of the *stakeholders' preferences analysis* is the fact that we have been able to identify "outliers", that is, stakeholders with very different preferences.

7.1.2 Optimal Solutions Analysis

The second contribution of this thesis aims to help decision-makers to better analyse and understand the differences and similarities between optimal solutions. We have seen that the Pareto optimal front from multi-objective search-based requirements selection and prioritisation techniques get very large and complex if the input consists of large numbers of requirements and stakeholders. This makes the task of decision makers who have to interpret these sets of optimal solutions very complicated. Although techniques exist to analyse those solutions, they do not help to decide on the design trade-offs and are therefore not very useful to requirements decision makers who are more concerned with design differences and similarities.

Our *optimal solutions analysis* technique aids decision makers in this task .We have used hierarchical agglomerative algorithms to group solutions on the Pareto optimal front according to design similarities that exist among them. Thus, instead of searching all the solutions for a particular one, the decision makers just have to first choose among design families and then focus on a specific solution within that family by comparing the solutions within it. We have used four types of visualisations to show how the design of solutions varies on the Pareto optimal front.

We have been able to find that close solutions on the Pareto optimal front (that is, solutions with similar objectives attainment) do not necessarily have similar design. This is also true in the opposite case; solutions far from each other Pareto optimal front can have very similar design. In the case of the next release problem, we have also seen that the cost of a solution does not always reflect the actual design of the solution as increasing cost of solutions do not always imply increasing the number of requirements included in the solutions.

7.2 Limitations and Future Work

We need to perform further validation in the field for our techniques. We need additional evaluation of the *optimal solutions analysis* to test to what extent our approach helps reducing the cognitive loads of decision makers and helps them identifying useful information about the solutions set that they could not identify otherwise. However, we are aware that the subjective nature of this exercise makes it hard to measure and compare. Similarly, we need further validation on the *stakeholders' preferences analysis* to further measure the extent to which it improves the final decision with respect to the stakeholders' preferences.

The tools presented in Matlab are currently restricted to only two dimensions (or two objectives) as it has been designed with the traditional cost-value approach in mind. Thus, for the *optimal solutions analysis* tool, the cluster distribution view on the Pareto front is optimised for these two dimensions. It can currently be easily extended to accommodate a third objective (in a third dimension). However, when we have more than 3 objectives, we cannot use this kind of visualization. Instead, we may need to adapt our technique to use other visualizations, such as heat maps as discussed in Chapter 2, when we have many objectives. However, the majority of requirement selection problems tend to look at two objectives - cost and value.

This thesis has presented how clustering can improve the requirements decision-making process. This work opens more avenues for further research.

During the *stakeholders' preferences analysis,* we have uncovered the groups of stakeholders according to their preferences. The next step will be to use stakeholder groups to find correlations between characteristics of the clusters of stakeholders and their preferences. Understanding why stakeholders rate requirements the way they do can be very useful during stakeholder analysis (Brugha and Varvasovszky 2000) and can help to uncover further requirements or identify ambiguities in requirements (Berry and Kamsties 2004). The stakeholder groups can be further used to determine the economic implications for the organisation commissioning the software. For example, for market-driven software the size of the groups can be used to determine the market share that could be reached if particular features are included in the software product and what potential revenue they can bring to the organisation.

The families of designs that have been identified during our *optimal solutions analysis* technique might be useful in the design of software product lines (Clements 2006). A software product line is a family of software that has a core common set of features and different optional features in each member of the family. Our optimal *solutions analysis* technique could be extended to help implement software product lines from the clusters that have been identified. Thus, once a family of design (a cluster of solutions in this case) has been identified using the *optimal solutions analysis*, we could enhance our technique to find the most efficient way of moving from one design to another by incrementally adding features.

Most of the applications of search-based optimisation in software engineering take as input extremely large volumes of data and produce Pareto fronts or sets of optimal solutions. Applying clustering on the input to group similar input together based on given characteristics can help to reduce the volume of the data being fed into the search algorithm. This can be especially useful when performing the search during real-time or on the fly in dynamic or adaptive systems where response time is important. The Pareto front generated in the different fields of search-base software engineering can also be clustered to provide important design insights to software engineers. In our work, we have looked at Pareto fronts generated for requirements engineering problems. This clustering technique can also be applied in other fields. For example, in search-based test case generation, it can be useful how the different test cases vary along the Pareto front before making any decisions. Of course, in each case, we will need to use the appropriate metrics to use to perform clustering.

Bibliography

Aittokoski, Timo, Sami Ayramo, and Kaisa Miettinen. 2009. "Clustering Aided Approach for Decision Making in Computationally Expensive Multiobjective Optimization." *Optimization Methods Software* 24: 157–174.

Alba, Enrique, and Francisco Chicano. 2007. "Ant Colony Optimization for Model Checking." In *Proceedings of the 11th International Conference on Computer Aided Systems Theory*, 523–530.

Baker, P, M. Harman, K Steinhofel, and A Skaliotis. 2006. "Search Based Approaches to Component Selection and Prioritization for the Next Release Problem." *22nd IEEE International Conference on Software Maintenance*: 176–185.

Berry, D.M., and E. Kamsties. 2004. "Ambiguity in Requirements Specification." In *Perspectives on Software Requirements*. Kluwer Academic Publishers.

Black, Paul E. 2006. "Manhattan Distance." Ed. Paul E Black. *Dictionary of Algorithms and Data Structures*. U.S. National Institute of Standards and Technology.

Blaikie, Norman. 2003. *Analyzing Quantitative Data: From Description to Explanation*. Sage Publications Ltd.

Boehm, B W, and P N Papaccio. 1988. *Understanding and Controlling Software Costs*. *IEEE Transactions on Software Engineering*. Vol. 14. IEEE Press.

Brin, S., and L. Page. 1998. "The Anatomy of a Large-scale Hypertextual Web Search Engine." Ed. Stanford University. *Computer Networks and ISDN Systems* 30 (1-7): 107–117.

Brugha, R, and Z Varvasovszky. 2000. "Stakeholder Analysis: a Review." *Health Policy and Planning* 15 (3): 239–246.

Castro-Herrera, C, J Cleland-Huang, and B Mobasher. 2009. "Enhancing Stakeholder Profiles to Improve Recommendations in Online Requirements Elicitation." In *17th IEEE International Requirements Engineering Conference*, 0:37–46.

Clements, Paul. 2006. *Software Product Lines*. Ed. Robert L Nord. *Software Product Lines*. Vol. 3714. Springer Berlin Heidelberg.

Crawshaw, Janet, and Joan Chambers. 2001. *A Concise Course in Advanced Level Statistics: With Worked Examples*. Nelson Thornes.

Cutting, Douglass R, David R Karger, Jan O Pedersen, and John W Tukey. 1992. "Scatter/Gather: a Cluster-based Approach to Browsing Large Document Collections." In *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ed. Nicholas Belkin, Peter Ingwersen, Annelise Mark Pejtersen, and Edward A Fox, 318–329. ACM.

Deb, K. 2001. *Multi-objective Optimization Using Evolutionary Algorithms (Wiley Interscience Series in Systems and Optimization)*. Wiley-Blackwell.

Deb, K, A Pratap, S Agarwal, and T Meyarivan. 2002. "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II." *IEEE Transactions on Evolutionary Computation* 6 (2): 182–197.

Duan, C., and J Cleland-Huang. 2007. "Clustering Support for Automated Tracing." In *Proceedings of the Twentysecond IEEEACM International Conference on Automated Software Engineering ASE 07*, 244. ACM Press.

Everitt, Brian S., Sabine Landau, and Morven Leese. 2001. *Cluster Analysis*. 4th ed. Hodder Arnold.

Feather, M. S., and T. Menzies. 2002. "Converging on the Optimal Attainment of Requirements." In *Proceedings IEEE Joint International Conference on Requirements Engineering*, 263–270. IEEE.

Finkelstein, A., M. Harman, SA Mansouri, J Ren, and Y. Zhang. 2009. "A Search Based Approach to Fairness Analysis in Requirement Assignments to Aid Negotiation, Mediation and Decision Making." *Requirements Engineering* 14 (4): 231–245.

Frank, Ronald Edward, William F. Massy, and Yoram Wind. 1972. *Market Segmentation*. Prentice-Hall.

Gan, Guojun, Chaoqun Ma, and Jianhong Wu. 2007. *Data Clustering: Theory, Algorithms, and Applications*. *ASASIAM Series on Statistics and Applied Probability*. Vol. 20. SIAM, Society for Industrial and Applied Mathematics.

Gay, Gregory, T. Menzies, Omid Jalali, Gregory Mundy, Beau Gilkerson, M. S. Feather, and James Kiper. 2010. "Finding Robust Solutions in Requirements Models." *Automated Software Engineering* 17 (1): 87–116.

Goldstein, H. 2005. "Who killed the virtual case file?." *IEEE SPECTRUM* 42 (9), 18.

Gower, J C. 1967. "A Comparison of Some Methods of Cluster Analysis." *Biometrics* 23 (4): 623–637.

Gower, J C. 1971. "A General Coefficient of Similarity and Some of Its Properties." *Biometrics* 27 (4): 857–871.

Hamming, R W. 1950. "Error Detecting and Error Correcting Codes." *Bell System Technical Journal* 29 (2): 147–160.

Harman, M, SA Mansouri, and Y Zhang. 2009. "Search Based Software Engineering: A Comprehensive Analysis and Review of Trends Techniques and Applications." In *TR-09-03*.

Harman, M. 2007. "The Current State and Future of Search Based Software Engineering." In *Future of Software Engineering (FOSE '07)*, 342–357. IEEE.

Harman, M., and B.F. Jones. 2001. “Search-based Software Engineering.” *Information and Software Technology* 43 (14): 833–839.

Harman, M., J Krinke, J Ren, and S Yoo. 2009. “Search Based Data Sensitivity Analysis Applied to Requirement Engineering.” *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation GECCO 09*: 1681–1688.

Hartigan, J. A., and M. A. Wong. 1979. “Algorithm AS 136: A K-Means Clustering Algorithm.” *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 28 (1): pp. 100–108.

Heaven, W., and E. Letier. 2011. “Simulating and Optimising Design Decisions in Quantitative Goal Models.” *2011 IEEE 19th International Requirements Engineering Conference*: 79–88.

Hubert, Lawrence J, and Joel R Levin. 1976. “A General Statistical Framework for Assessing Categorical Clustering in Free Recall.” *Psychological Bulletin* 83 (6): 1072–1080.

Jaccard, P. 1908. “Nouvelles Recherches Sur La Distribution Florale.” *Bulletin De La Société Vaudoise Des Sciences Naturelles.* (44): pp.223–270.

Jain, Anil K., and Richard C. Dubes. 1988. *Algorithms for Clustering Data*. Prentice Hall College Div.

Kramer, Joseph, Sunil Noronha, and John Vergo. 2000. “A User-centered Design Approach to Personalization.” *Communications of the ACM* 43 (8): 44–48.

Lamsweerde, Axel Van. 2009a. *Requirements Engineering: From System Goals to UML Models to Software Specifications*. John Wiley & Sons.

Lamsweerde, Axel Van. 2009b. “Reasoning About Alternative Requirements Options.” Ed. Alexander Borgida, Vinay K Chaudhri, Paolo Giorgini, and Eric S K Yu. *Conceptual Modeling Foundations and Applications* 5600: 380–397.

Larsen, Bjornar, and Chinatsu Aone. 1999. “Fast and Effective Text Mining Using Linear-time Document Clustering.” Ed. Surajit Chaudhuri and David Madigan. *Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining KDD 99* 5 (5): 16–22.

Lauesen, Soren. 2002. *Software Requirements: Styles & Techniques*. Addison-Wesley Professional.

Laurent, P, and J Cleland-Huang. 2009. “Lessons Learned from Open Source Projects for Facilitating Online Requirements Processes.” In *Proceedings of the 15th International Working Conference on Requirements Engineering: Foundation for Software Quality*, 5512:240–255. Springer-Verlag.

Laurent, P, J Cleland-Huang, and C. Duan. 2007. "Towards Automated Requirements Triage." In *15th IEEE International Requirements Engineering Conference RE 2007*, 131–140. IEEE.

Likert, R. 1932. "A Technique for the Measurement of Attitudes." *Archives of Psychology* 22 (140): 55.

Lim, S. L. 2010. "Social Networks and Collaborative Filtering for Large-Scale Requirements Elicitation." University of New South Wales, Sydney, Australia.

Lim, S. L., D. Damian, F. Ishikawa, and A. Finkelstein. 2012. "Using Web 2.0 for Stakeholder Analysis: StakeSource and Its Application in Ten Industrial Projects." In *Managing Requirements Knowledge*, ed. Walid Maalej and Anil Kumar Thurimella. In press. Springer Computer Science Editorial.

Lim, S. L., D. Quercia, and A. Finkelstein. 2010. "StakeSource: Harnessing the Power of Crowdsourcing and Social Networks in Stakeholder Analysis." In *ACM/IEEE 32nd International Conference on Software Engineering*, 239 –242.

Ma, Y., and C. Zhang. 2008. "Quick Convergence of Genetic Algorithm for QoS-driven Web Service Selection." *Computer Networks* 52 (5) (April 1): 1093–1104.

Mancoridis, S., B.S. Mitchell, Y. Chen, and E.R. Gansner. 1999. "Bunch: a Clustering Tool for the Recovery and Maintenance of Software System Structures." In *Proceedings IEEE International Conference on Software Maintenance - 1999 (ICSM'99). "Software Maintenance for Business Change" (Cat. No.99CB36360)*, 50–59. IEEE.

Mancoridis, S., B.S. Mitchell, C. Rorres, Y. Chen, and E.R. Gansner. 1998. "Using Automatic Clustering to Produce High-level System Organizations of Source Code." In *Proceedings. 6th International Workshop on Program Comprehension. IWPC'98 (Cat. No.98TB100242)*, 45–52. IEEE Comput. Soc.

Mattson, Christopher A, Anoop A Mullur, and Achille Messac. 2004. "Smart Pareto Filter: Obtaining a Minimal Representation of Multi-objective Design Space." *Engineering Optimization* 36 (6): 721–740.

Mccall, Chester H. 2001. "An Empirical Examination of the Likert Scale: Some Assumptions, Development and Cautions." *80th Annual CERA Conference* 54: 1–11.

McMinn, Phil. 2004. "Search-based Software Test Data Generation: a Survey." *Software Testing Verification and Reliability* 14 (2): 105–156.

McQuitty, Louis L. 1966. "Similarity Analysis by Reciprocal Pairs for Discrete and Continuous Data." *Educational and Psychological Measurement* 26 (4) (December): 825 –831.

Milligan, Glenn, and Martha Cooper. 1985. "An Examination of Procedures for Determining the Number of Clusters in a Data Set." *Psychometrika* 80 (2): 159–179.

Milne, Alastair, and Neil Maiden. 2011. “Power and Politics in Requirements Engineering: A Proposed Research Agenda.” In *2011 IEEE 19th International Requirements Engineering Conference*, 187–196. IEEE.

Mojena, R. 1977. “Hierarchical Grouping Methods and Stopping Rules: An Evaluation.” *The Computer Journal* 20 (4): 359–363.

Morse, J N. 1980. “Reducing the Size of the Nondominated Set: Pruning by Clustering.” *Computers & Operations Research* 7 (1-2): 55–66.

Noyes, Jan, and Chris Baber. 1999. *User-Centred Design of Systems*. Springer.

Nuseibeh, Bashar, and Steve Easterbrook. 2000. “Requirements Engineering: a Roadmap.” In *Proceedings of the Conference on The Future of Software Engineering - ICSE ’00*, 35–46. ACM Press.

Obayashi, Shigeru, and Daisuke Sasaki. 2003. “Visualization and Data Mining of Pareto Solutions Using Self-Organizing Map.” *Evolutionary MultiCriterion Optimization* 2632: 796–809.

Page, L., S. Brin, Rajeev Motwani, and Terry Winograd. 1998. “The PageRank Citation Ranking: Bringing Order to the Web.” *World Wide Web Internet And Web Information Systems* 54 (2): 1–17.

Parnas, D L. 1979. “Designing Software for Ease of Extension and Contraction.” *IEEE Transactions on Software Engineering* SE-5 (2): 128–138.

Pryke, Andy, Sanaz Mostaghim, and Alireza Nazemi. 2007. “Heatmap Visualization of Population Based Multi Objective Algorithms.” In *Proceedings of the 4th International Conference on Evolutionary Multi-criterion Optimization*, 361–375. Springer-Verlag.

Raden, David. 1985. “Strength-Related Attitude Dimensions.” *Social Psychology Quarterly* 48 (4): 312–330.

Rand, William M. 1971. “Objective Criteria for the Evaluation of Clustering Methods.” *Journal of the American Statistical Association* 66 (336): 846–850.

Reformat, M., X. Chai, and J. Miller. 2007. “On the Possibilities of (pseudo-) Software Cloning from External Interactions.” *Soft Computing* 12 (1) (June 30): 29–49.

Reformat, M., and J. Miller. 2003. “Experiments in Automatic Programming for General Purposes.” In *Proceedings. 15th IEEE International Conference on Tools with Artificial Intelligence*, 366–373. IEEE Comput. Soc.

Robertson, Suzanne, and James C. Robertson. 2006. *Mastering the Requirements Process*. Addison Wesley.

Rosenman, M. A., and J. S. Gero. 1985. “Reducing the Pareto Optimal Set in Multicriteria Optimization(With Applications to Pareto Optimal Dynamic Programming).” *Engineering Optimization* 8 (3): 189–206.

Rousseeuw, Peter. 1987. “Silhouettes: a Graphical Aid to the Interpretation and Validation of Cluster Analysis.” *Journal of Computational and Applied Mathematics* 20 (1): 53 – 65.

Räihä, O., K. Koskimies, and E. Mäkinen. 2008. *Simulated Evolution and Learning*. Ed. Xiaodong Li, Michael Kirley, Mengjie Zhang, David Green, Vic Ciesielski, Hussein Abbass, Zbigniew Michalewicz, et al. Vol. 5361. Berlin, Heidelberg: Springer Berlin Heidelberg.

Saaty, Thomas L. 1980. *Analytic Hierarchy Process.* McGraw-Hill International Book Co.

Schilit, B, N Adams, and R Want. 1994. “Context-aware Computing Applications.” In *Proceedings., Workshop on Mobile Computing Systems and Applications.*, 85–90. IEEE Comput. Soc. Press.

Seyff, Norbert, Gregor Ollmann, and Manfred Bortenschlager. 2011. “iRequire: Gathering End-user Requirements for New Apps.” In *2011 IEEE 19th International Requirements Engineering Conference*, 347–348. IEEE.

Siegel, S. 1957. “Nonparametric Statistics.” *American Statistician* 11 (3): 13–19.

Sneath, P.H.A. 1957. “The Application of Computers to Taxonomy.” *Journal of General Microbiology* 17 (1): 201–206.

Sokal, R. R., and C. D. Michener. 1958. “A Statistical Method for Evaluating Systematic Relationships.” *University of Kansas Science Bulletin* 38: 1409–1438.

Sokal, R. R., and J. F. Rohlf. 1962. “The Comparison of Dendrograms by Objective Methods.” *Taxon* 11 (2): 33–40.

Stevens, S. S. 1951. “Mathematics, Measurement, and Psychophysics.” In *Handbook of Experimental Psychology*, ed. S. S. Stevens, 1–49. New York: John Wiley & Sons.

Sørensen, T. 1948. “A Method of Establishing Groups of Equal Amplitude in Plant Sociology Based on Similarity of Species and Its Application to Analyses of the Vegetation on Danish Commons.” *Biologiske Skrifter* 5: 1–34.

Tukey, John W. 1977. *Exploratory Data Analysis*. Addison Wesley.

Veerappa, V., and E. Letier. 2011. “Understanding Clusters of Optimal Solutions in Multi-Objective Decision Problems.” In *Proceedings of the 2011 IEEE 19th International Requirements Engineering Conference*, 89–98. IEEE Computer Society.

Ward, Jr. 1963. “Hierarchical Grouping to Optimize an Objective Function.” *Journal of the American Statistical Association* 58: 236–244.

Wedel, M., and W. A. Kamakura. 1999. *Market Segmentation: Conceptual and Methodological Foundations (International Series in Quantitative Marketing)*. Springer.

Wiegers, Karl. 2003. *Software Requirements 2*. Microsoft Press.

Yoo, Shin, and Mark Harman. 2007. "Pareto Efficient Multi-objective Test Case Selection." In *Proceedings of the 2007 International Symposium on Software Testing and Analysis - ISSTA '07*, 140. New York, New York, USA: ACM Press.

Zave, P. 1995. "Classification of Research Efforts in Requirements Engineering." In *Proceedings of 1995 IEEE International Symposium on Requirements Engineering RE95*, 315–321. ACM.

Zhang, Y. 2010. "Multi-Objective Search-based Requirements Selection and Optimisation". King's College London, UK.

Zhang, Y., A. Finkelstein, and M. Harman. 2008. "Search Based Requirements Optimisation: Existing Work and Challenges." In *Proceedings of the 14th International Conference on Requirements Engineering Foundation for Software Quality*, ed. Barbara Paech and Colette Rolland, 5025:88–94. SPRINGER-VERLAG BERLIN.

Zhang, Y., and M. Harman. 2010. "Search Based Optimization of Requirements Interaction Management." In *2nd International Symposium on Search Based Software Engineering*, 47–56. IEEE.

Zhang, Y., M. Harman, and S. L. Lim. 2012. "Empirical Evaluation of Search Based Requirements Interaction Management." *Information and Software Technology* In press.

Zhao, Ying, and George Karypis. 2002. "Evaluation of Hierarchical Clustering Algorithms for Document Datasets." In *Proceedings of the Eleventh International Conference on Information and Knowledge Management - CIKM '02*, ed. Charles Nicholas, David Grossman, Konstantinos Kalpakis, Sajda Qureshi, Han Van Dissel, and Len Seligman, 515. New York, New York, USA: ACM Press.

Appendix A - Tool Support for Stakeholders' Preferences Analysis

We have implemented a tool to enable decision makers to perform *stakeholders' preferences analysis* in Matlab. The aim of this tool is to facilitate the *stakeholders' preferences analysis* by providing a simple easy to use interface. Although the tool provides some statistical analysis features, it is not a comprehensive list and the results can be exported to a more powerful statistical tool to perform more enhanced analyses. Figure 66 gives an overview of how the tool works.

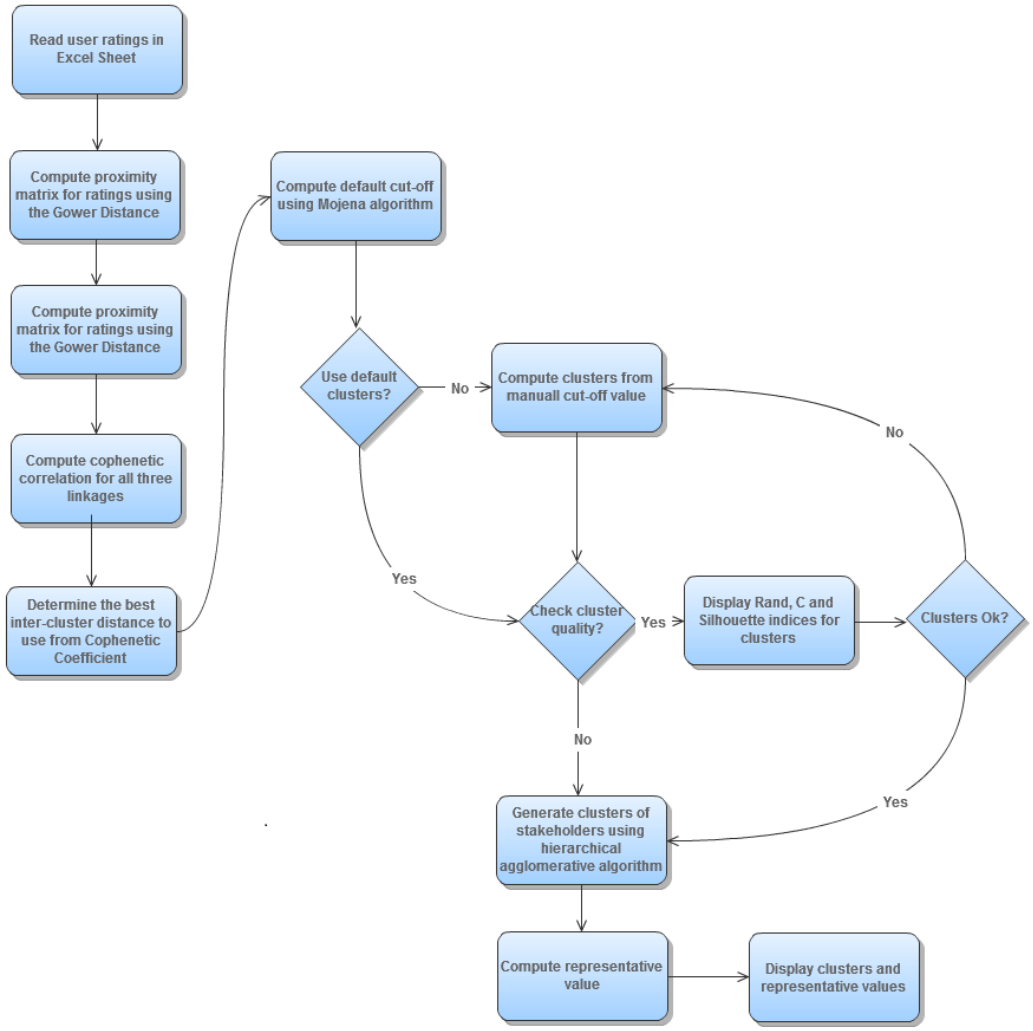


Figure 66 Flow chart for tool - Stakeholders' Preferences Analysis

Tool Graphical User Interface Overview

The interface is a simple GUI that presents all information from a single screen as shown in Figure 67.

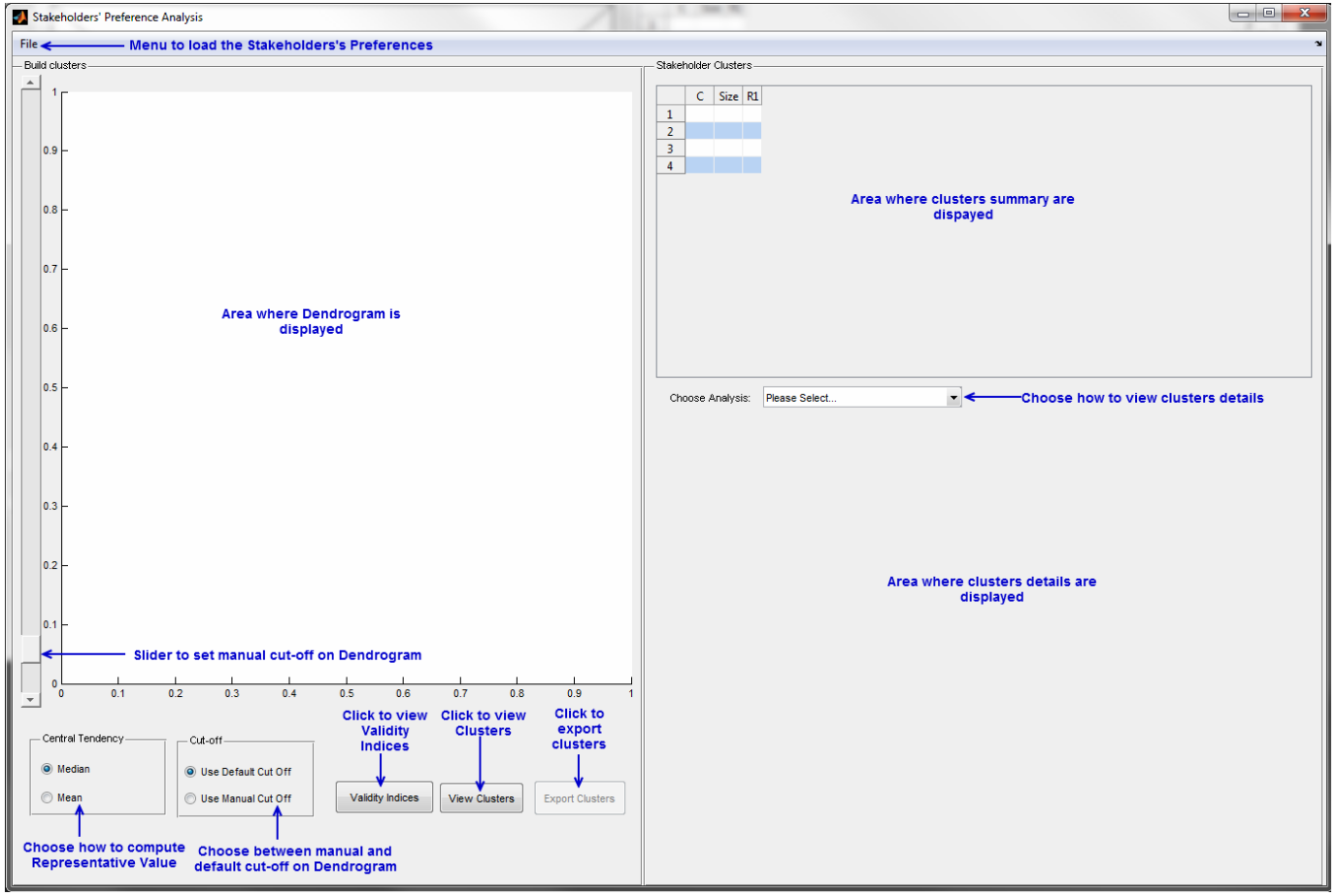


Figure 67 Stakeholders' Preferences Analysis Tool Overview

The GUI consists of four main areas:

1. Dendrogram Display

   In this area of the tool, we view the dendrogram that results from the ratings. The decision maker can also use the controls here to set the manual cut-off for the dendrogram.
2. Clusters Generation Parameters Display

   This is the interface where the decision maker chooses type of measure of central tendency will be used to compute the representative preferences for the clusters and which cut-off to use. He can also access the controls to view the quality indices, export the clusters overview and launch the actual generation of the clusters from here.
3. Clusters Summary Display

   This area of the tool displays summary information about the clusters.
4. Analyses Display

   Decision makers can use the options on this area of the tool to further analyse the clusters of stakeholders.

Generating Clusters of Stakeholders

To generate the clusters of stakeholders, decision makers using our tool need to perform the following steps.

Load the Excel spread sheet where the stakeholder preference is saved.

The spreadsheet must contain three sheets: one named "Ratings" which will contain the ratings of the stakeholders, one name "Costs" with the costs of the requirements and the other named "Stakeholders" which will contain the stakeholders' identifiers. The "Ratings" sheet will have as first row the labels of the requirements. Each row after the title row represents the ratings for one stakeholder. The "Costs" sheet will contain only one row, where each column represents the cost for the corresponding requirement for that column from the "Ratings" sheet. If there is no cost information, the cost must be set to 0. The "Stakeholders" sheet will contain only one column with each row representing the stakeholders in the same order as their preference in the "Ratings" sheet.

To load the spread sheet, click on the "Load" option from the "File" menu on the tool as illustrated in Figure 68 Loading the spread sheet to read Stakeholders' preferences.

This will open a file browser dialog. Select the required file and click on the "Open" button on the dialog box. Clicking this button will make the tool load the data in the spread sheet in Matlab's working memory and generate the dendrogram for the dataset using the hierarchical clustering algorithm. The tool also automatically determines which linkage to use by computing and the Cophenetic Correlation coefficient for each of the three candidate linkages.

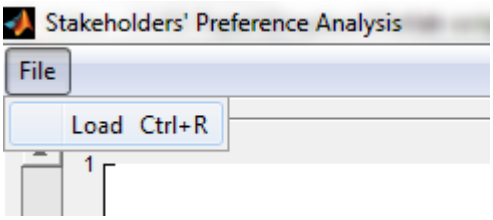


Figure 68 Loading the spread sheet to read Stakeholders' preferences

The tool then displays the generated dendrogram on the Dendrogram display area as shown in Figure 69. It also computes the default (Mojena's) cut-off that it displays on the dendrogram as a black dotted horizontal line to give the decision makers how the clusters will be.

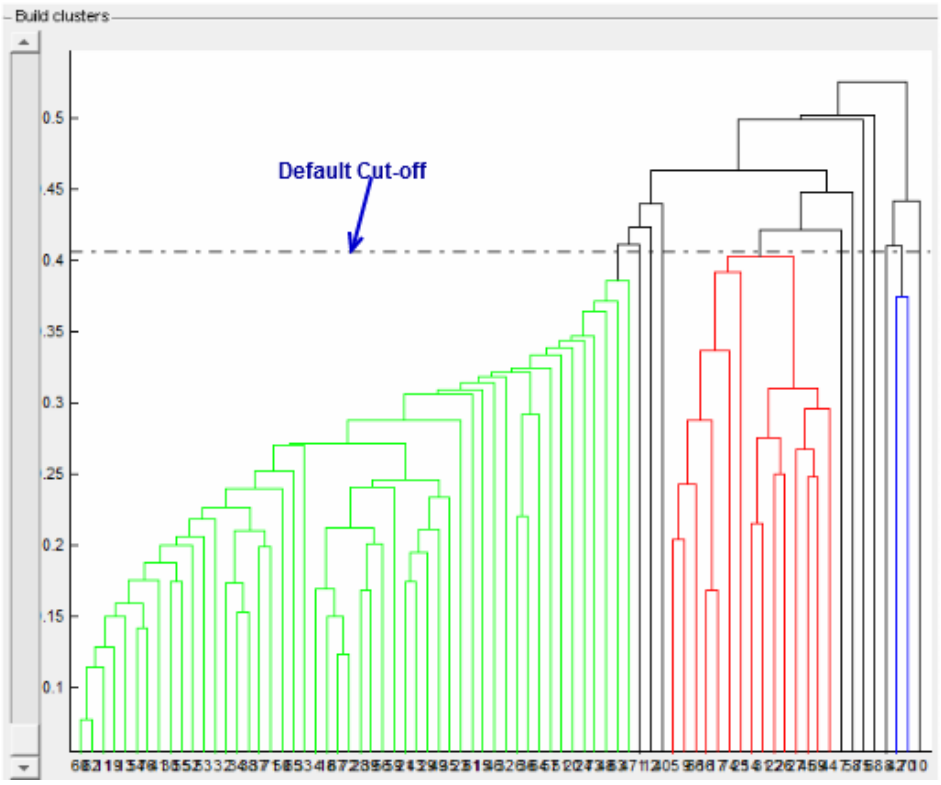


Figure 69 Generated Dendrogram with Default Cut-off

If the decision maker is not happy with the default cut-off, he can use the slider on the left of the Dendrogram display area to manually set the cut-off. The manual cut-off line is depicted by a second red horizontal line on the dendrogram. This is illustrated in Figure 70.

Decision maker may want to check the clusters' quality for the clusters being generated at a specific cut-off. He can access the cluster quality indices by clicking on the "Validity" indices button in the Clusters Generation Parameters Display area (shown in Figure 71).

This will load the cluster indices pop up screen in Figure 72 that displays the Rand, C and Silhouette indices for all possible numbers of clusters for the data. The tool will compute these indices from two clusters to N clusters where N is the number of stakeholders being analysed.

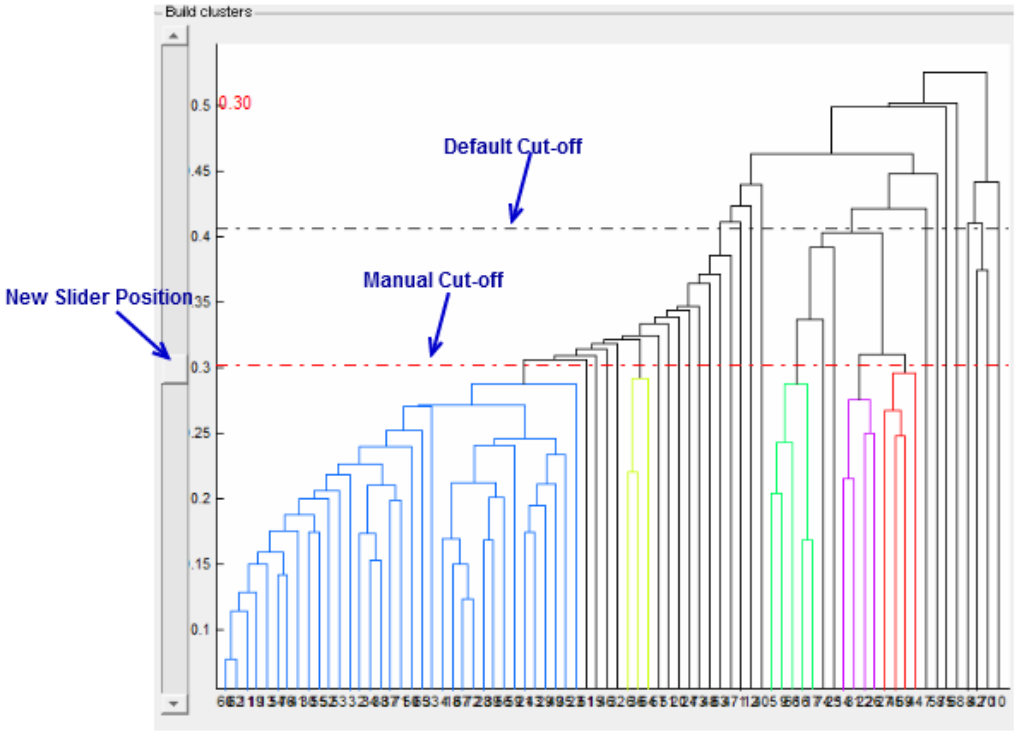


Figure 70 Using the Manual Cut-off Slider

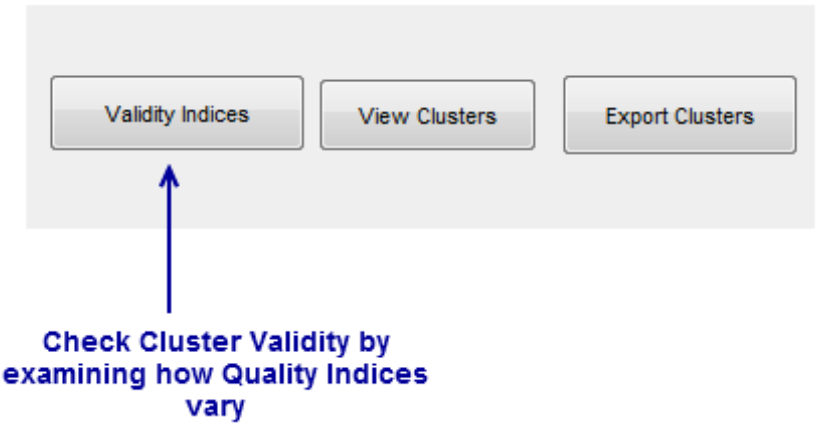


Figure 71 Control to view cluster quality indices

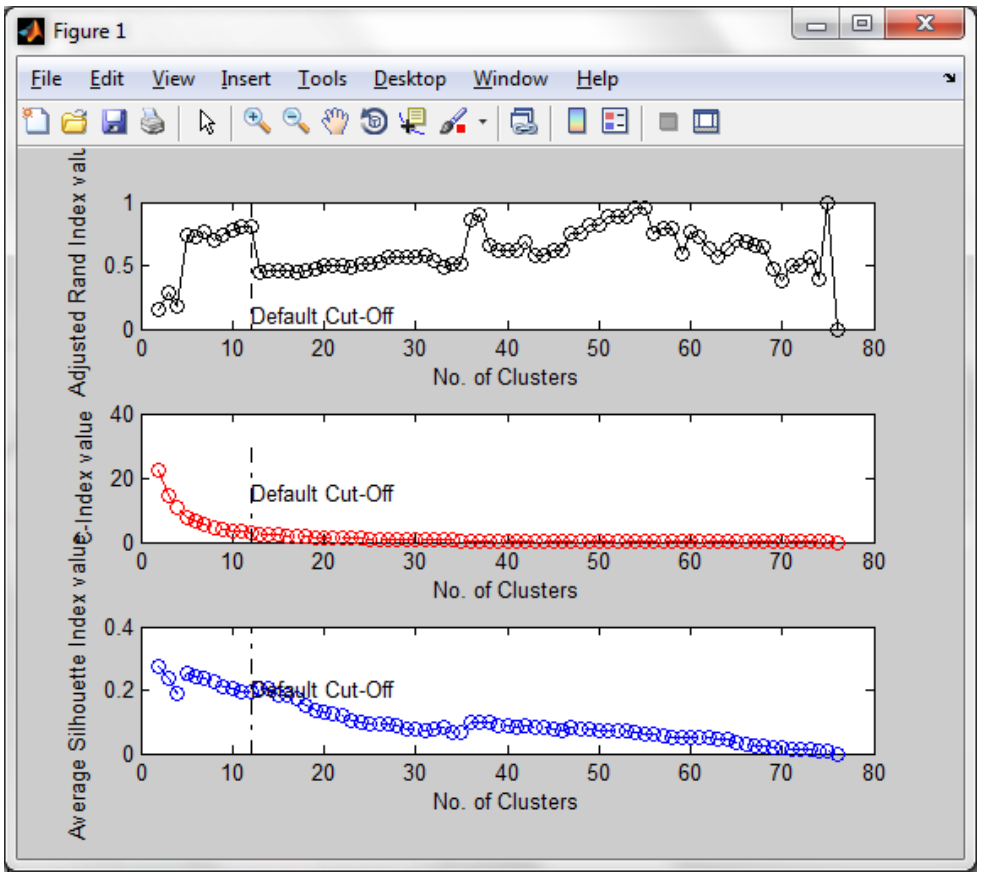


Figure 72 Quality Indices for Clusters

Choose the measure of central tendency and cut-off to use.

The decision maker must then choose whether to use the mean or median rating for the clusters as representative value for the clusters. Another parameter that he must

choose is which cut-off he wishes to use – manual or default. He can access the required controls from the Clusters Generation Parameters Display of the tool (Figure 73).

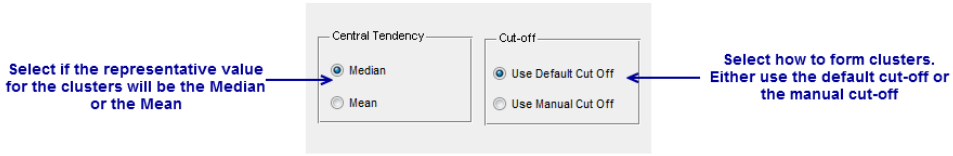


Figure 73 Cluster generation parameters controls

Generate clusters

Once the decision maker is happy that he has properly set the parameters for the cluster generations, he clicks on the "View Clusters" button from the Clusters Generation Parameters Display of the tool shown in Figure 74. This instructs the tool to form the clusters based on the selected cut-off value and compute the cluster preferences for each of the requirements for the clusters.

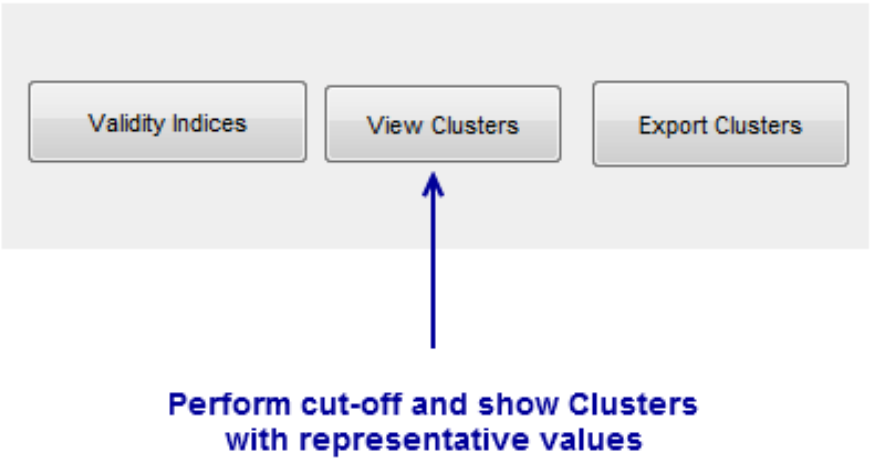


Figure 74 Control to generate Clusters

This populates the Clusters Summary Display area of the tool with the clusters and summary information about the clusters as shown in Figure 75.

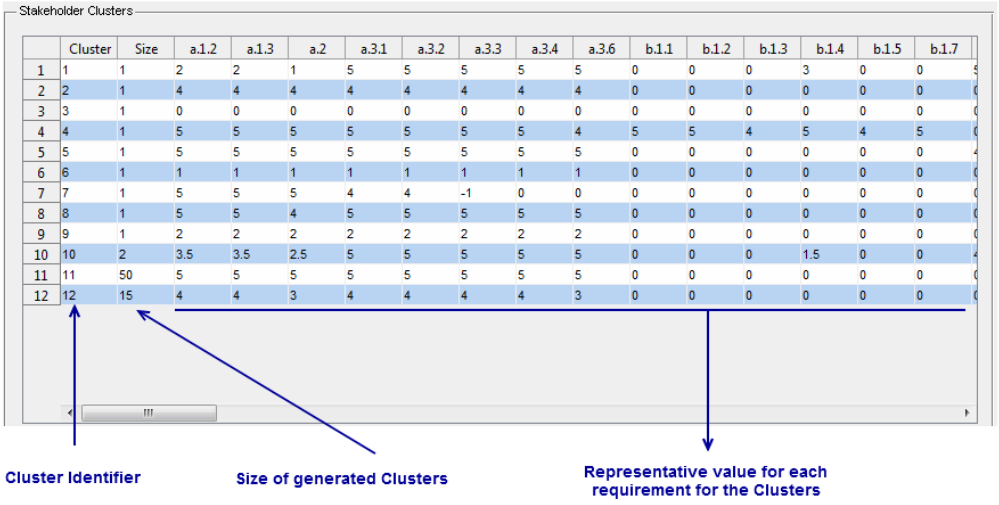


Figure 75 Populated Cluster Summary Area

Export clusters' summary to Excel

The decision maker may want to export the cluster summary information to Excel to view later or to use in decision-making tools. To do so, he needs to click on the "Export Clusters" button on the Clusters Generation Parameters Display of the tool (Figure 76). If there are requirements for which our technique cannot find a cluster preference, the tool opens a dialog box prompting the decision maker for the value to use in those cases as shown in Figure 77.

Figure 76 Control to export Clusters' Summary to Excel

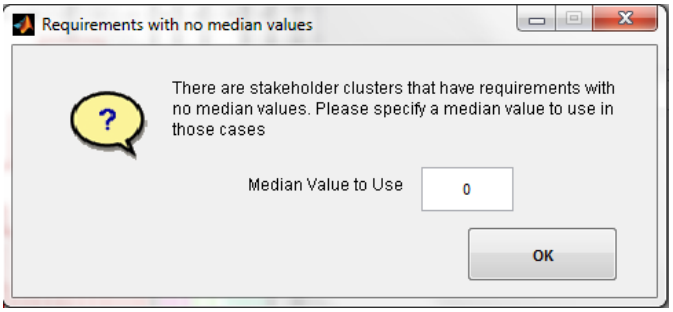


Figure 77 Dialog to input values for requirements with no median values

Further Analysis of Clusters

Once the tool has discovered the clusters of stakeholders, the decision maker may further want to analyze the clusters. He can do so by accessing the dropdown list from the Analyses Display area of the tool as shown in Figure 78.

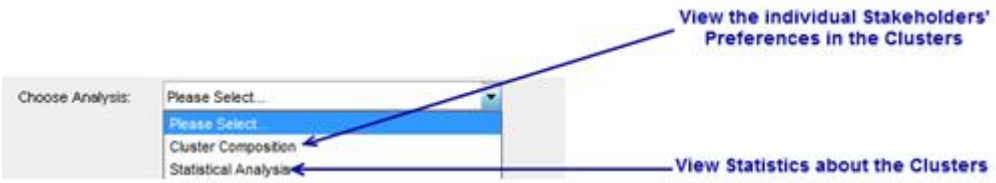
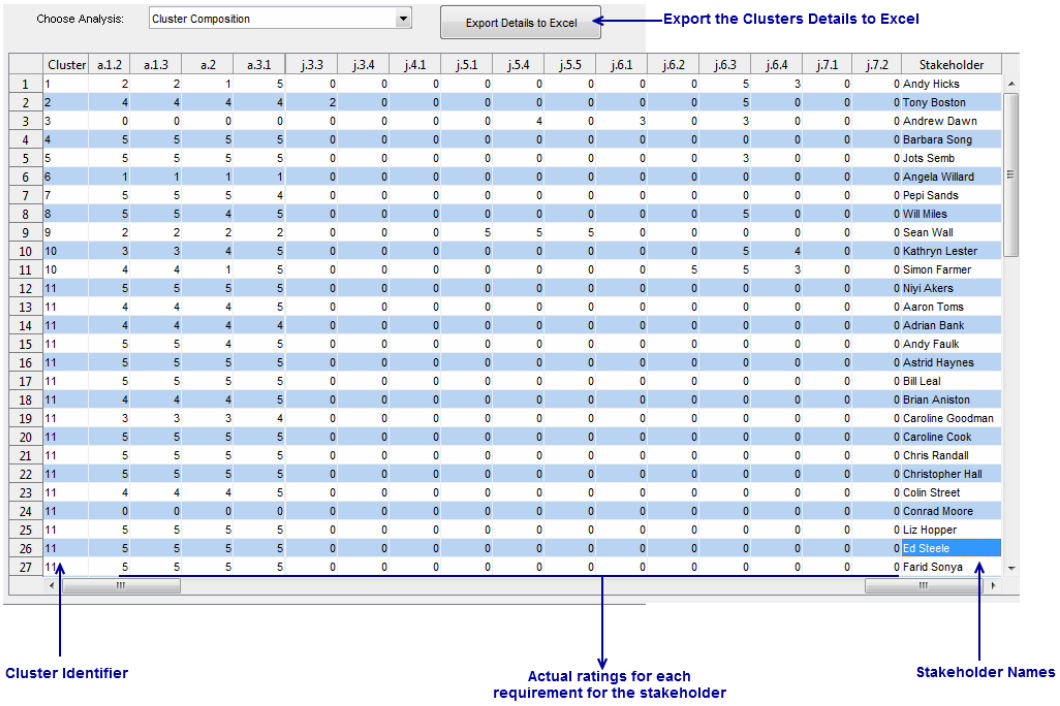


Figure 78 Analysis Options for Clusters

Our tool currently provides 2 types of analyses that can be performed. These are described next.

Cluster Composition Analysis

This option enables the decision maker to view which stakeholder is in which cluster. This information is displayed in tabular form as shown in Figure 79. If the decision maker wants to save this information for later analyses, he can do so by clicking on the "Export Details to Excel" button.



	Cluster	a.1.2	a.1.3	a.2	a.3.1	j.3.3	j.3.4	j.4.1	j.5.1	j.5.4	j.5.5	j.6.1	j.6.2	j.6.3	j.6.4	j.7.1	j.7.2	Stakeholder
1	1	2	2	1	5	0	0	0	0	0	0	0	0	5	3	0	0	Andy Hicks
2	2	4	4	4	4	2	0	0	0	0	0	0	0	5	0	0	0	Tony Boston
3	3	0	0	0	0	0	0	0	0	4	0	3	0	3	0	0	0	Andrew Dawn
4	4	5	5	5	5	0	0	0	0	0	0	0	0	0	0	0	0	Barbara Song
5	5	5	5	5	5	0	0	0	0	0	0	0	0	3	0	0	0	Jots Semb
6	6	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	Angela Willard
7	7	5	5	5	4	0	0	0	0	0	0	0	0	0	0	0	0	Pepi Sands
8	8	5	5	4	5	0	0	0	0	0	0	0	0	5	0	0	0	Will Miles
9	9	2	2	2	2	0	0	0	5	5	5	0	0	0	0	0	0	Sean Wall
10	10	3	3	4	5	0	0	0	0	0	0	0	0	5	4	0	0	Kathryn Lester
11	10	4	4	1	5	0	0	0	0	0	0	0	5	5	3	0	0	Simon Farmer
12	11	5	5	5	5	0	0	0	0	0	0	0	0	0	0	0	0	Niyi Akers
13	11	4	4	4	5	0	0	0	0	0	0	0	0	0	0	0	0	Aaron Toms
14	11	4	4	4	4	0	0	0	0	0	0	0	0	0	0	0	0	Adrian Bank
15	11	5	5	4	5	0	0	0	0	0	0	0	0	0	0	0	0	Andy Faulk
16	11	5	5	5	5	0	0	0	0	0	0	0	0	0	0	0	0	Astrid Haynes
17	11	5	5	5	5	0	0	0	0	0	0	0	0	0	0	0	0	Bill Leal
18	11	4	4	4	5	0	0	0	0	0	0	0	0	0	0	0	0	Brian Aniston
19	11	3	3	4	4	0	0	0	0	0	0	0	0	0	0	0	0	Caroline Goodman
20	11	5	5	5	5	0	0	0	0	0	0	0	0	0	0	0	0	Caroline Cook
21	11	5	5	5	5	0	0	0	0	0	0	0	0	0	0	0	0	Chris Randall
22	11	5	5	5	5	0	0	0	0	0	0	0	0	0	0	0	0	Christopher Hall
23	11	4	4	4	5	0	0	0	0	0	0	0	0	0	0	0	0	Colin Street
24	11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Conrad Moore
25	11	5	5	5	5	0	0	0	0	0	0	0	0	0	0	0	0	Liz Hopper
26	11	5	5	5	5	0	0	0	0	0	0	0	0	0	0	0	0	Ed Steele
27	11	5	5	5	5	0	0	0	0	0	0	0	0	0	0	0	0	Farid Sonya

Figure 79 Cluster Details

Statistical Analysis

When the decision maker chooses this option, the tool presents a control for the decision maker to select one or more clusters to view the box plots of ratings per requirement for each cluster Figure 80. After selecting the required clusters, he need to click on the "OK" button and the box plots are displayed on a popup screen as in Figure 81. The requirements are sorted in order of costs. If a cluster has only one stakeholder in it, the tool only plots the ratings instead of a box plot.

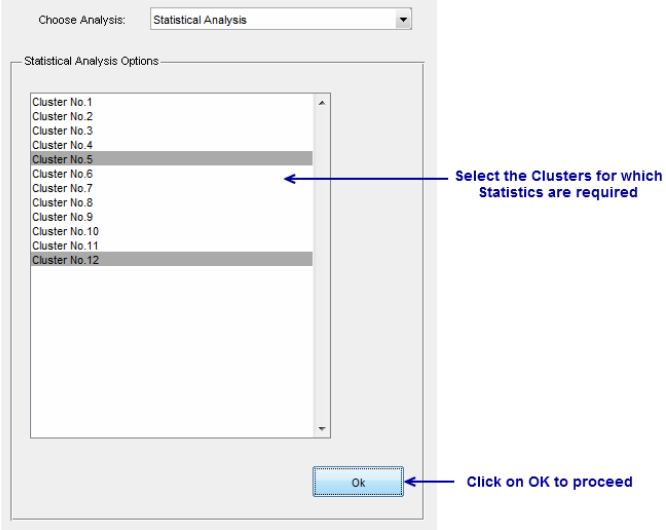


Figure 80 Cluster Selection for Statistical Analysis

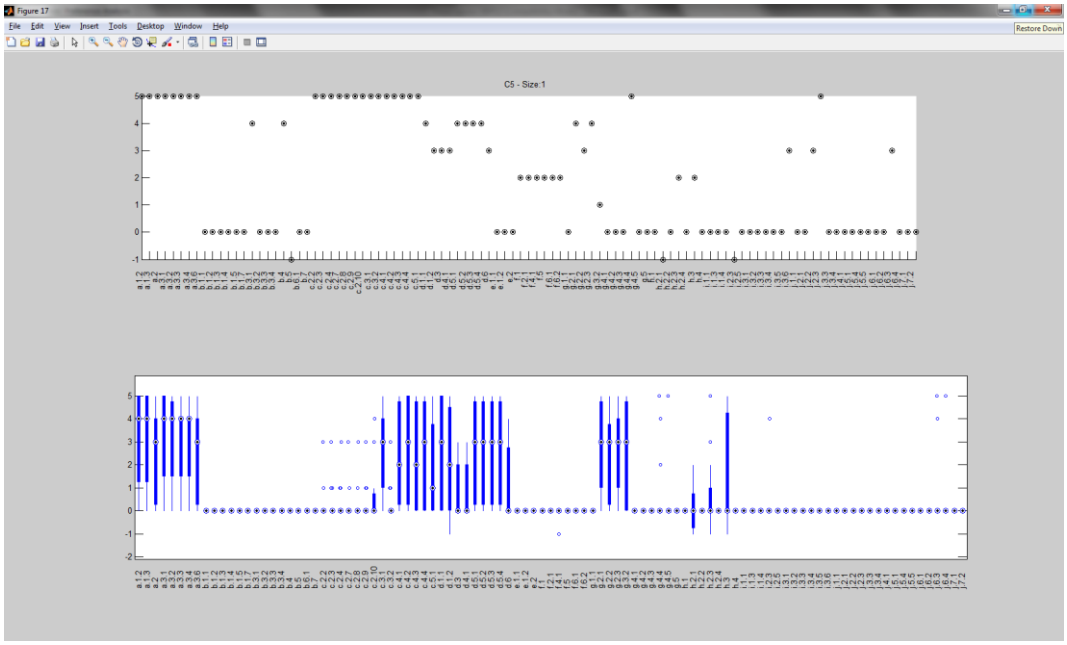


Figure 81 Boxplot for Selected Clusters

Appendix B - Tool Support for Optimal Solutions Analysis

We have implemented a tool to enable decision makers to perform *optimal solutions analysis* in Matlab. The aim of this tool is to facilitate the *optimal solutions analysis* by providing a simple easy to use interface. The tool provides the visualizations discussed in this thesis to help identify the similarities and differences among optimal solutions in the context of cost-value multi-objective next release problems. Figure 82 gives an overview of how the tool works.

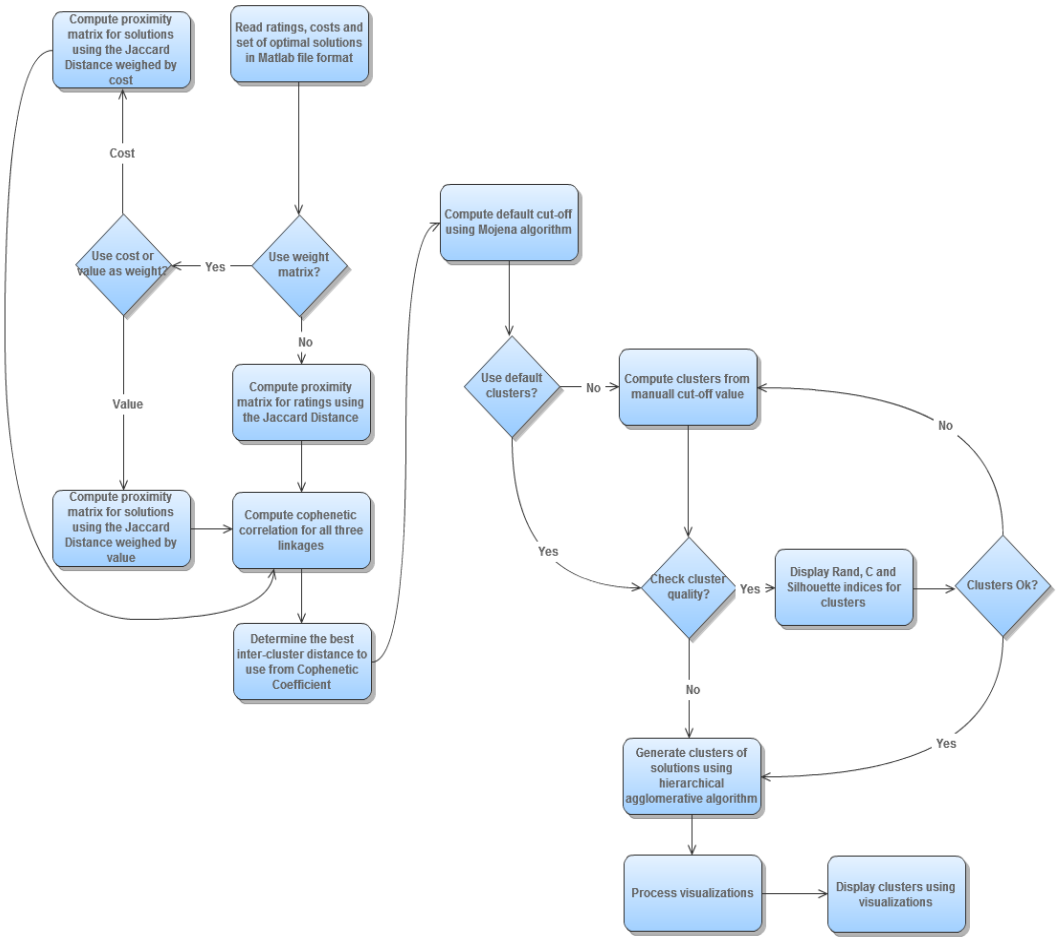


Figure 82 Flow chart for tool - Optimal Solutions Analysis

Tool Graphical User Interface

The interface is a simple GUI that presents all information from a single screen as shown in Figure 83.

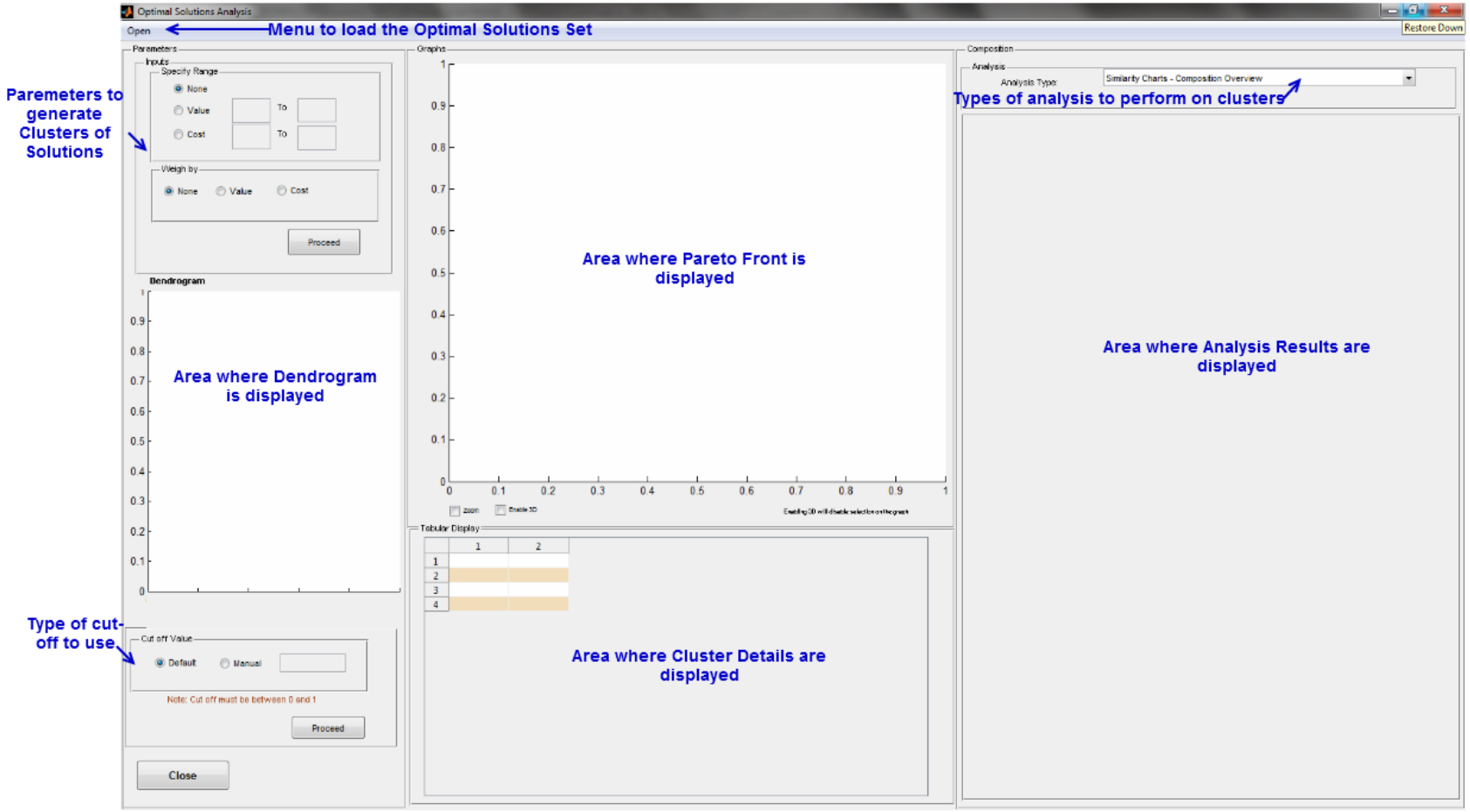


Figure 83 Optimal Solutions Analysis Tool Overview

The GUI consists of six main areas:

1. Cluster Parameters

   The controls on this part of the tool enable the decision maker to provide information about the ranges of value or cost on which he wants to do the analysis. He can also specify if he wants to use any weighting when measuring the distance between the solutions here.

2. Dendrogram Display

   In this area of the tool, we view the dendrogram that results from the solutions.

3. Cut-off Parameters

   The decision maker uses the controls here to choose the type of cut-off for the dendrogram. If he selects the manual cut-off option, he has to specify a cut-off distance.

4. Cluster Details Display

   This area of the tool displays details information about the clusters' composition.

5. Cluster Distribution of the Pareto Front Display

   This area of the tool displays how clusters are distributed along the Pareto front.

6. Analyses Display

   The decision maker can use the options on this area of the tool to further analyze and generate other views for the clusters of solutions.

Generating Clusters of Solutions

To generate the clusters of solutions, decision makers using our tool need to perform the following steps.

Load the Matlab .mat file where the Pareto optimal solutions are saved.

Since our work based on the multi-objective search algorithms developed in (Zhang 2010), our input file is the .mat file generated in from them. The .mat file must contain three vectors. One vector named "R" that contains the values of the requirements elicited from stakeholders and the costs of these requirements. Each column of this vector represents a requirement while each row will be the preferences for a stakeholder or stakeholder group, the exception being the last row that contains the costs of the

requirements. A second vector named “C” that contains the weights or importance of the stakeholders. There is only one row in this vector and the columns will each represent a stakeholder or stakeholder groups. A last vector called “best_pareto” that contains the Pareto optimal solutions. Each row of this vector will be a solution. Each column will represent a requirement, with exception of the two last ones which represent cost and value of the solution.

To load the .mat file, click on the “Load” option from the “File” menu on the tool as illustrated in Figure 84. This will open a file browser dialog. Select the required file and click on the “Open” button on the dialog box. Clicking this button will make the tool load the data in Matlab’s working memory.

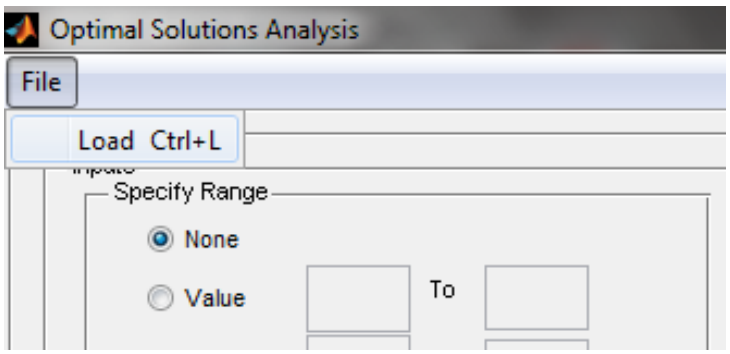


Figure 84 Loading .mat file in tool

Set the parameters required to generate the clusters of solutions

Before generating the clusters of solutions, the decision maker needs to tell the tool if he wants to generate the clusters only for a specific cost or value range on the Pareto optimal front. By default the tool considers all the solutions in the optimal set. Specifying a range here will make the tool cluster only the solutions in that range. This may be useful if the decision maker knows beforehand what cost or value range he is looking to investigate. The decision maker also needs to specify if he wants the tool to use the normal Jaccard distance or the weighted Jaccard distance. Currently, our tool allows the user to choose the cost or value as the weight matrix. The controls our tools provide to make these selections are illustrated in Figure 85.

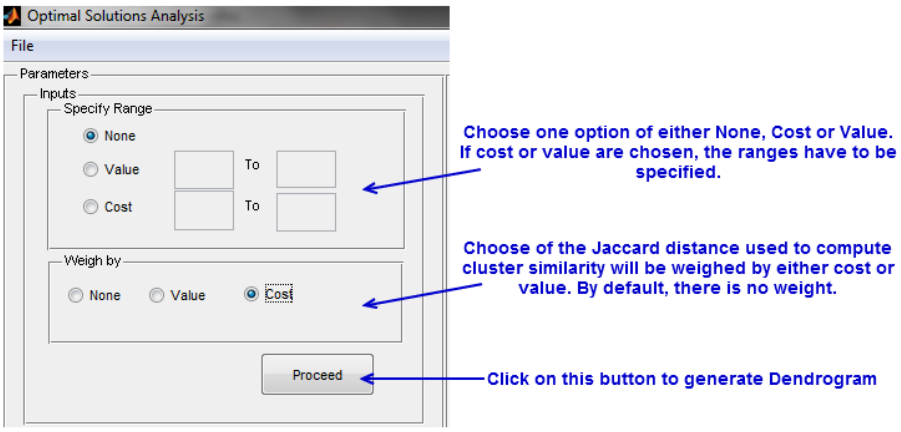


Figure 85 Setting parameters to generate Clusters

When the decision maker clicks on the button "Proceed", our tool generates the dendrogram for the dataset using the hierarchical clustering algorithm. The tool also automatically determines which linkage to use by computing and the Cophenetic Correlation coefficient for each of the three candidate linkages. The dendrogram is displayed in the area designated for it in the tool GUI. The tool also computes the default cut-off that it shows on the dendrogram as a black dotted horizontal line to give the decision makers how the clusters will be. An example of such a dendrogram in the tool is shown in Figure 86.

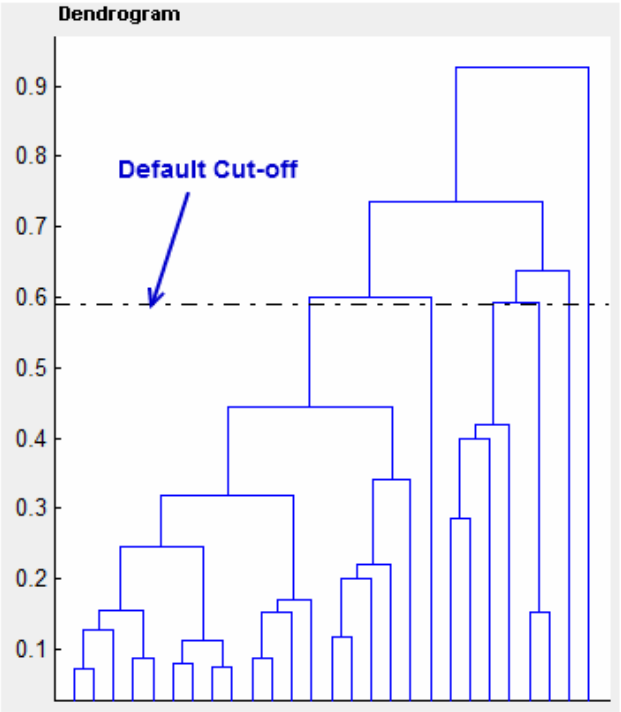


Figure 86 Example of Dendrogram with default cut-off in tool

If the decision maker is not satisfied with the clusters that are formed using the default cut-off, he can specify a manual cut-off value for the dendrogram using the controls the tool provides for this purpose. These are shown in Figure 87. When he clicks on the "Proceed" button, our tool computes the clusters and displays their summary information in the cluster details display area on the tool. Figure 88 shows an example of the display of the cluster summary information.

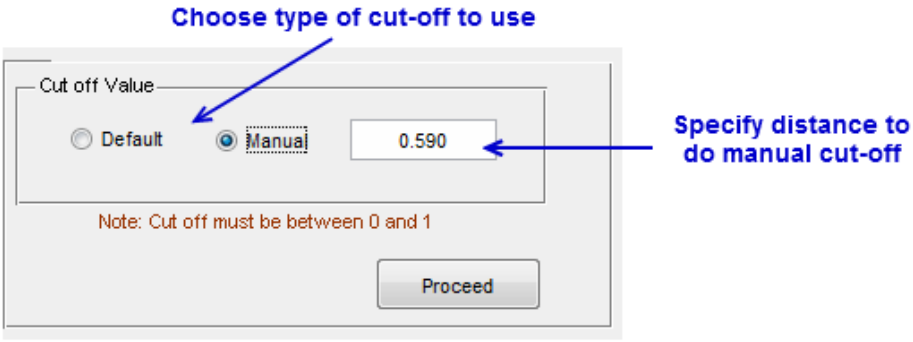
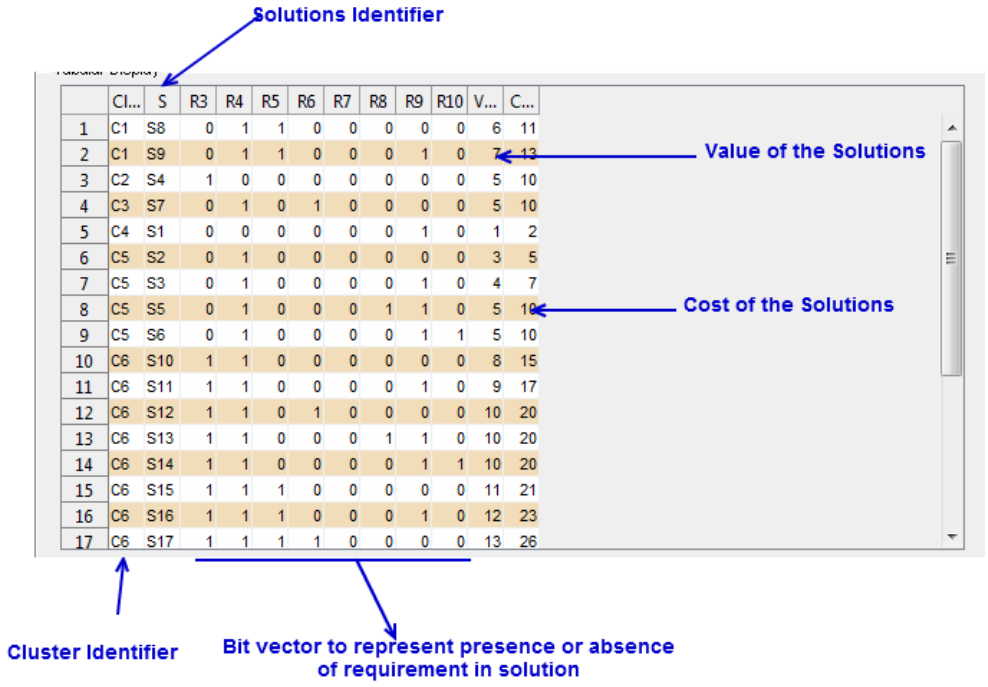


Figure 87 Choosing cut-off value



	Cl...	S	R3	R4	R5	R6	R7	R8	R9	R10	V...	C...
1	C1	S8	0	1	1	0	0	0	0	0	6	11
2	C1	S9	0	1	1	0	0	0	1	0	7	13
3	C2	S4	1	0	0	0	0	0	0	0	5	10
4	C3	S7	0	1	0	1	0	0	0	0	5	10
5	C4	S1	0	0	0	0	0	0	1	0	1	2
6	C5	S2	0	1	0	0	0	0	0	0	3	5
7	C5	S3	0	1	0	0	0	0	1	0	4	7
8	C5	S5	0	1	0	0	0	1	1	0	5	10
9	C5	S6	0	1	0	0	0	0	1	1	5	10
10	C6	S10	1	1	0	0	0	0	0	0	8	15
11	C6	S11	1	1	0	0	0	0	1	0	9	17
12	C6	S12	1	1	0	1	0	0	0	0	10	20
13	C6	S13	1	1	0	0	0	1	1	0	10	20
14	C6	S14	1	1	0	0	0	0	1	1	10	20
15	C6	S15	1	1	1	0	0	0	0	0	11	21
16	C6	S16	1	1	1	0	0	0	1	0	12	23
17	C6	S17	1	1	1	1	0	0	0	0	13	26

Figure 88 Cluster Summary

Visualizations

Our tool provides multiple views of the clusters of solutions to enable the decision maker to assess the design similarities among the solutions on the Pareto optimal front.

Apart from the distribution of clusters on the Pareto front, all other views are accessed from Analysis area of the tool GUI as shown in Figure 89. The views selected from that control are displayed in Analyses display of our tool.

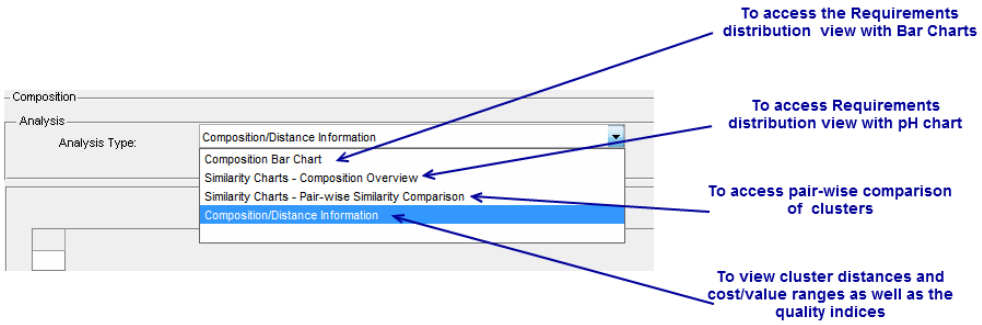


Figure 89 Cluster analysis options

We next describe the visualizations available in the tool.

Distribution of clusters on the Pareto front.

This view is generated as soon as the clusters are formed when the decision maker has clicked on the "Proceed" button. The points representing the individual solutions in the same cluster are of the same colour. When the clusters have more than one solution in them, their boundary is outlined with an ellipse with the same colour as the points in them. An example of this view on the designated area on the tool is shown in Figure 90.

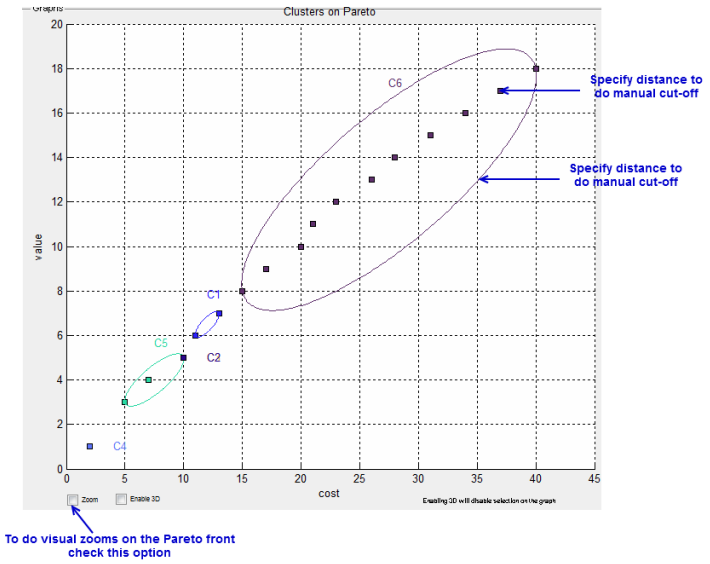


Figure 90 Distribution of clusters on the Pareto Optimal front visualization

Our tool provides the option to visually zoom on an area of the Pareto front to have a more close up view of the clusters in that area. This can be achieved by checking the "Zoom" option below the Pareto Front (Figure 90). Our tool also enables the decision maker to rotate the clusters in 3 dimensions. The first two dimensions are the value and the cost of the solutions. The third dimension is the distance of each cluster from the cluster labelled "C1" in the dataset. This functionality can be activated by checking the "Enable 3D" option under the Pareto front. Figure 91 shows an instance of the 3D view in our tool.

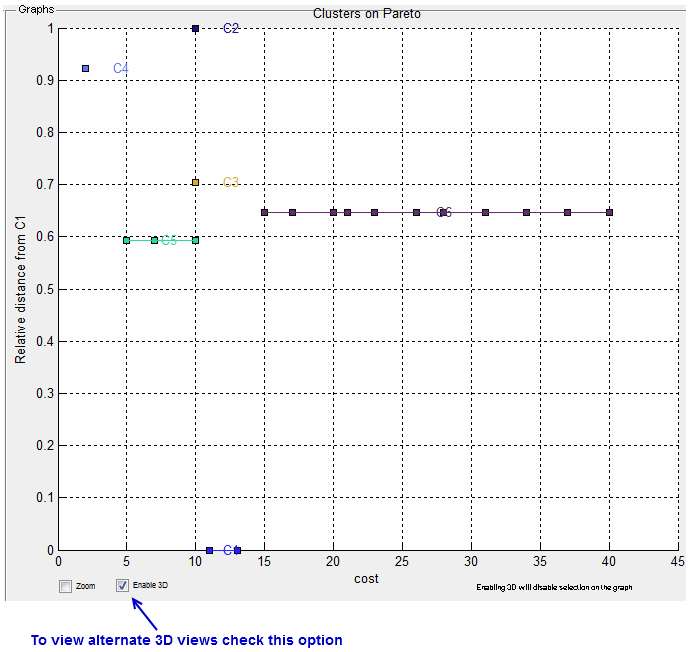


Figure 91 3D rotation of Cluster Distribution View

Requirements Distribution per Cluster – Bar Chart

In this view, the tool plots bar charts (in increasing cost order of clusters) to show the decision maker the number of solutions in each cluster that contain a given requirement. The y axes of the charts show the number of solutions containing the requirement and the x- axes label the requirements. This is illustrated in Figure 92.

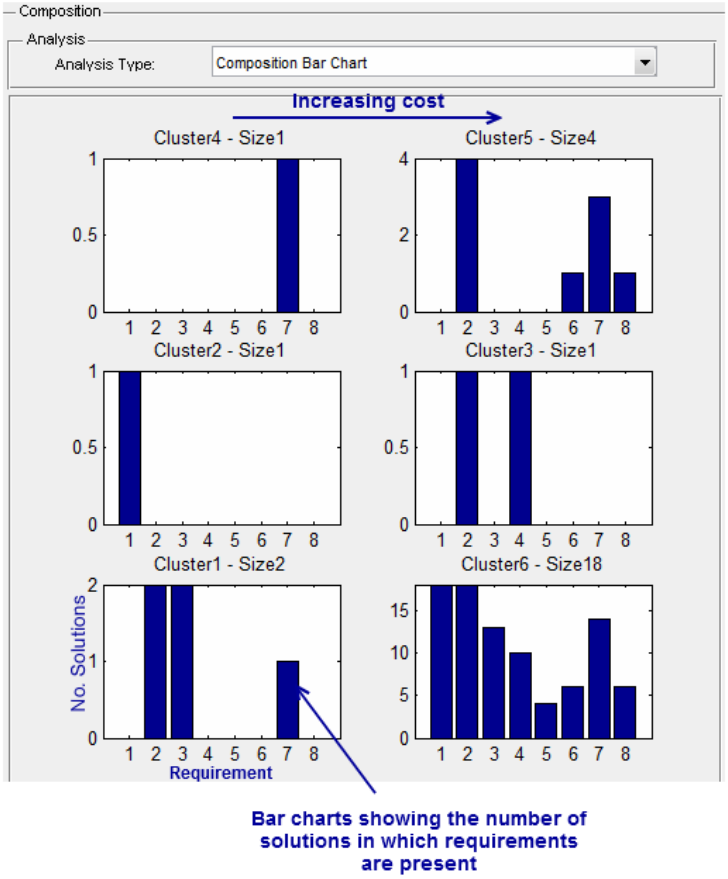


Figure 92 Cluster Composition - Bar Chart View

Requirements Distribution per Cluster- pH Chart

This version of the requirements distribution view is a colour chart with cells with different shades for each requirement for each cluster. The clusters are again displayed in increasing cost order. Each shade of the cell represents a range of percentage composition of the cluster for that requirement. For example, the darkest shade for a requirement indicates that all i.e. 100% solutions in that cluster contain that requirement. The next lighter shade for a requirement indicates that 90-99% of the solutions in that cluster contain that requirement and so on until we reach the lightest shade which shows that no solution contain that requirement. The cells are also labelled "All" if it represents 100% presence, "None" if represents 0% presence (or absence) and "Some" for any value between. Figure 93 shows an example of this view in our tool.

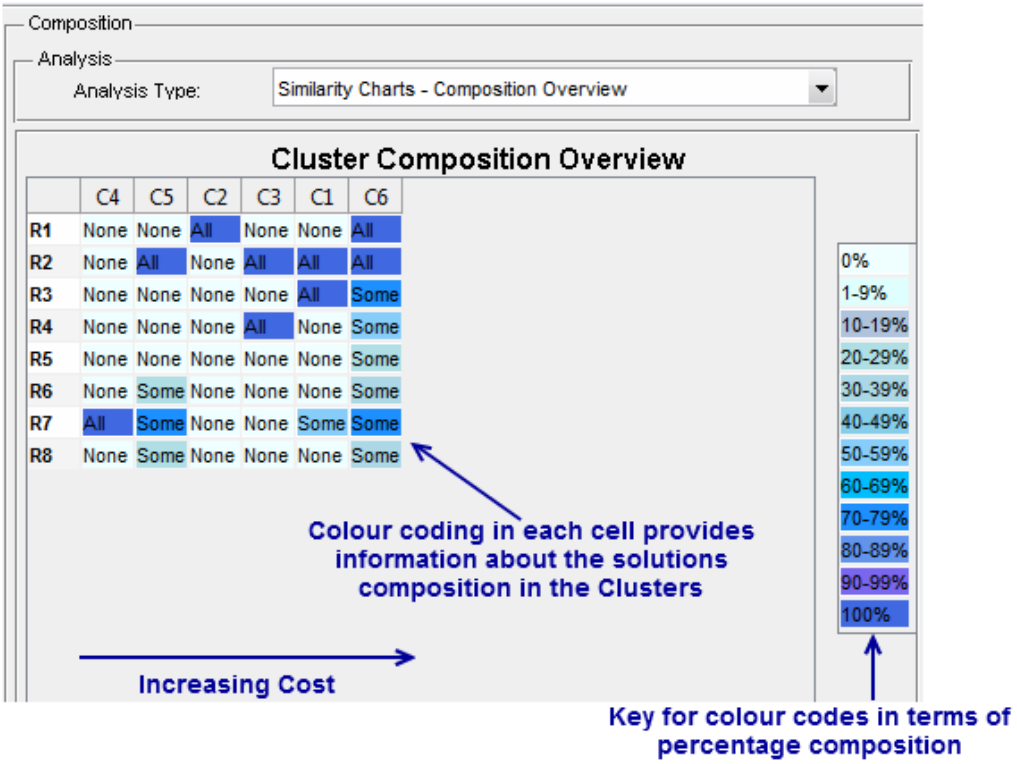


Figure 93 Cluster Composition - pH Chart

Pair-wise Comparison of Clusters

The decision maker may need to further compare two specific clusters of solutions to check how similar they are. Our tool enables him to choose two clusters and then displays which requirements are in all, some or none of the solutions in the clusters. He can further compare the individual solutions in the clusters if he wants to in this view. The pair-wise comparison of clusters views in our tool are illustrated in Figure 94.

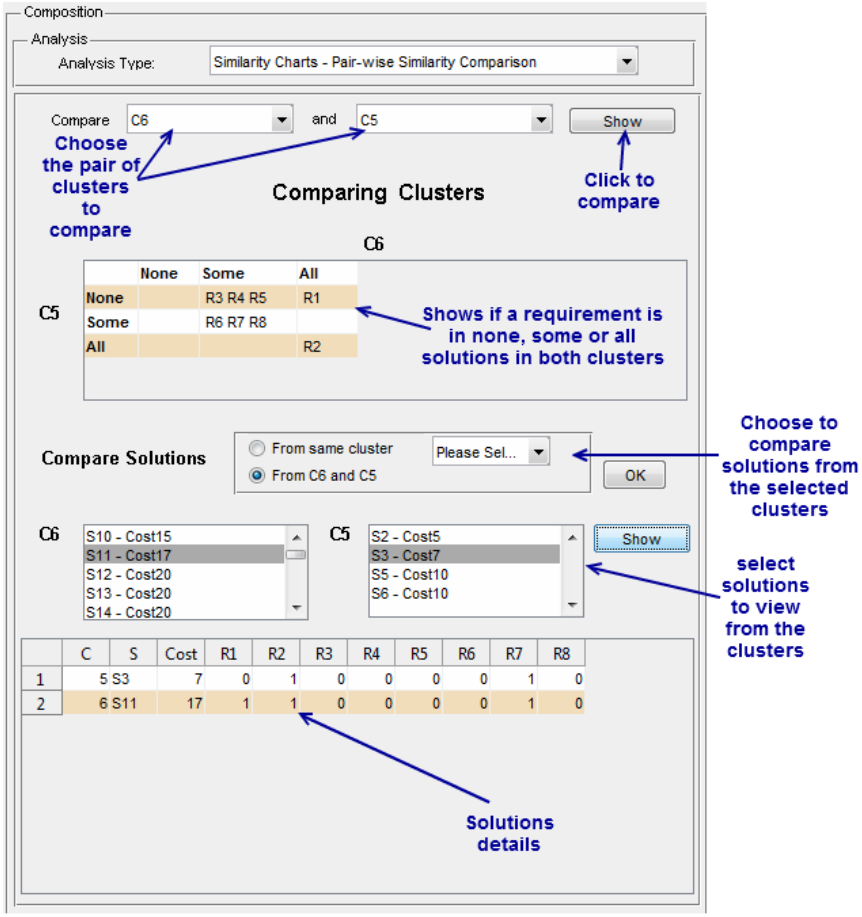
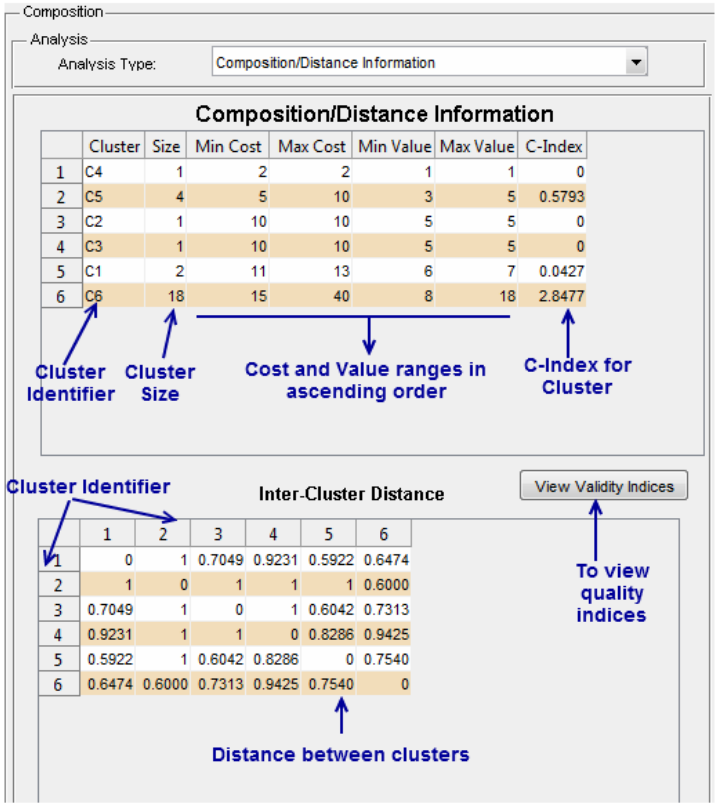


Figure 94 Pair-wise Comparison of Clusters View

Composition Distance Information

This view in our tool helps the decision maker to check the quality of the clusters. It shows the range of the cost and values for each cluster as well as their C-indices. It also displays the inter-cluster distances between the clusters. The decision-maker can view the quality indices for the clusters by clicking on the "View Validity Indices" button on the GUI. This will load the cluster indices pop up screen in Figure 96 that displays the Rand, C and Silhouette indices for all possible numbers of clusters for the data. The tool will compute these indices from two clusters to N clusters where N is the number of solutions being analysed. The composition/distance information for the clusters is displayed on the screen as shown in Figure 95.



	Cluster	Size	Min Cost	Max Cost	Min Value	Max Value	C-Index
1	C4	1	2	2	1	1	0
2	C5	4	5	10	3	5	0.5793
3	C2	1	10	10	5	5	0
4	C3	1	10	10	5	5	0
5	C1	2	11	13	6	7	0.0427
6	C6	18	15	40	8	18	2.8477

	1	2	3	4	5	6
1	0	1	0.7049	0.9231	0.5922	0.6474
2	1	0	1	1	1	0.6000
3	0.7049	1	0	1	0.6042	0.7313
4	0.9231	1	1	0	0.8286	0.9425
5	0.5922	1	0.6042	0.8286	0	0.7540
6	0.6474	0.6000	0.7313	0.9425	0.7540	0

Figure 95 Composition/Distance View

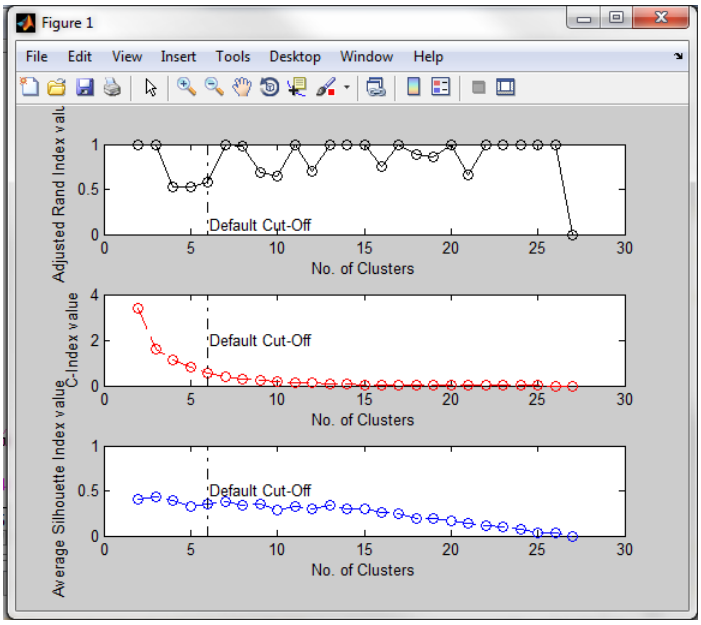


Figure 96 Quality Indices

Appendix C - Detailed Cluster Composition for RALIC Stakeholders

Cluster	Size	a.1.2	a.1.3	a.2	a.3.1	a.3.2	a.3.3	a.3.4	a.3.6	b.1.1	b.1.2	b.1.3	b.1.4	b.1.5	b.1.7	b.3.1	b.3.2	b.3.3	b.3.4
1	**4**	4	4	4	4	4	4	4	4										5
2	**5**	5	5	5	5	5	5	5	5										
3	**3**	3	3	2	4	4	4	4	4										
4	**1**	0	0	0	0	0	0	0	0										
5	**4**	1	1	1	1	1	1	1	1										
6	**1**	5	5	5	4	4	-1	0											
7	**17**	4	4	4	5	4	4	4	4		5	5	4			5			
8	**38**	5	5	5	5	5	5	5	5	5	5	4	4.5	3.5	5	4	5	5	
9	**3**	3	3	1	5	5	5	5	5				3			4.5			

Cluster	Size	b.4	b.5	b.6.1	b.7	c.2.2	c.2.3	c.2.4	c.2.7	c.2.8	c.2.9	c.2.10	c.3.1	c.3.2	c.4.1	c.4.2	c.4.3	c.4.4	c.5.1
1	**4**					5	5	5	5	5	5	5	5	5	5	5	5	5	5
2	**5**					3	3	3	3	3	3	3	3	3	4	5	4	5	4
3	**3**					0	0	0	0	0	0	0	1	0	0	0	0	0	0
4	**1**					5	5	5	5	5	5	5	5	5	5	5	5	5	5
5	**4**					1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5	1.5
6	**1**												0		-1	-1	-1	-1	0
7	**17**	5	2			4	4	4	4	4	4	4	4	4	4	4	4	4	4
8	**38**	4	-1	5	3	5	5	5	5	5	5	5	5	5	5	5	5	5	5
9	**3**	4.5	2			1	1	1	1	1	1	1	4		2	2	2.5	2	2.5

Cluster	Size	d.1.1	d.1.2	d.3	d.4.1	d.5.1	d.5.2	d.5.3	d.5.4	d.6	e.1.1	e.1.2	e.2	f.1	f.2.1	f.4.1	f.5	f.6.1	f.6.2
1	**4**	3	2	2	2	5	5	5	5	2									
2	**5**	5	4	3	3	5	5	5	5	3									
3	**3**	0	0	0	0	0	0	0	0	0									
4	**1**	5	5	5	5	5	5	5	5	5									
5	**4**	2	2	2	2	2.5	2.5	2.5	2.5	2.5	3	3	3	5	5	2	5	5	5
6	**1**	5	1																
7	**17**	4	4	4	4	4	4	4	4	4	4	4	4	3.5	3.5	3.5	3.5	4	3.5
8	**38**	5	5	5	5	5	5	5	5	5	3	4	4	3.5	4.5	4.5	3.5	4.5	4
9	**3**	5	0.5	4	1	4.5	3	3	3	1.5	4	4	4	5	5	5	5	5	5

Cluster	Size	g.1.1	g.2.1	g.2.2	g.2.3	g.3.2	g.4.1	g.4.2	g.4.3	g.4.4	g.4.5	g.5	h.1	h.2.1	h.2.2	h.2.3	h.2.4	h.3	h.4
1	**4**		-1	-1	-1	3				4			2	2	2	3	2	4	2
2	**5**		5	2	5	3								-1		-1		-1	
3	**3**		1	1	1	0								0		0		0	
4	**1**		3	3	3	3								0		0		0	
5	**4**		2	2	2	2				2	5	5	2	1	2	1	2	1	2
6	**1**		-1	-1	-1	2								-1		-1		-1	
7	**17**		4	3	4	4	3		3	4.5	5		5	0	3	3		3	
8	**38**	4.5	4	4	4.5	4		4		5	5		5	0	4	2	4.5	3	4.5
9	**3**		4.5	4.5	4.5	2.5				5			2	0.5	2	2	2	1.5	2

Cluster	Size	i.1.1	i.1.3	i.1.4	i.2.3	i.2.5	i.3.1	i.3.2	i.3.3	i.3.4	i.3.5	i.3.6	j.1.1	j.2.1	j.2.2	j.2.3	j.3.3	j.3.4	j.4.1
1	**4**												5				2		
2	**5**																		
3	**3**																		
4	**1**																		
5	**4**				4							5	5		5	5			
6	**1**																		
7	**17**				3				4			4			0	3			
8	**38**	5	4	4	1.5	5	5	5	4	4	3.5	4	5	4	4	5		5	2
9	**3**				1							4.5		3	4	5			

Cluster	Size	j.5.1	j.5.4	j.5.5	j.6.1	j.6.2	j.6.3	j.6.4	j.7.1	j.7.2
1	**4**						4.5	5		
2	**5**									
3	**3**									
4	**1**									
5	**4**	5	5	5			5			
6	**1**									
7	**17**	4	4	4			5	4		
8	**38**	5	4	5	3		4		5	5
9	**3**					5	5	3		