ISSN (Print) : 0974-6846
Indian Journal of Science and Technology, *Vol 9(40), DOI: 10.17485/ijst/2016/v9i40/98938, October 2016* ISSN (Online) : 0974-5645

An Automated Requirement Ranking Approach for Identifying Software Requirement

M. Narendhar[1*] and K. Anuradha[2]

[1]Department of CSE, SNTI, JNTUH, Kukatpally, Hyderabad - 500085, Telangana, India; mnarender53@gmail.com
[2]Department of CSE, GRIET, JNTUH, Kukatpally, Hyderabad - 500085, Telangana, India; kodali.anuradha@yahoo.com

Abstract

Objective: This paper presents an automated requirement ranking approach for identifying software requirements, which combines the project requirements of the order of approximation calculated through learning techniques. **Method:** We proposed an algorithm RRA, which automatically calculates, approximate ranks for the requirements based on priority rules. **Findings:** The algorithm considers requirements as inputs and outputs the best suitable requirements for software development in three stages namely Pairing of requirements, Extraction of priority and Learning of Priority. The proposed algorithm is more effective compared to CBRank, especially for more number of extracted pairs. **Improvement:** The work can be extended to produce accurate ranks for more number of extracted requirement pairs.

Keywords: Automation, Requirement Ranking Process, Software Requirement

1. Introduction

Software requirements priority plays a key role in the development of software, especially software that allows planning for the public, combining strategy for budgeting and preparation, as well as market scheme[1,2]. In fact, it is measured to be a difficult task for multi-criteria and decision-making process. The latest procedures[(3-5etc...)] are simple models for this process and are shared among, in defining the target model for ordering, requirements specification attributes to encode the selected model. Acquisition of certain values for these attributes for all requirements are to be considered and the composition of the scale reserves attributes associated with the target standards.

The ranking is based on the assumption that the policy analysis, evaluation of the situation and the conditions under which they can be defined independently on the nature of the current set of multiple ranking criteria. Which cannot be put together. Therefore, the value calculated from the ranks of the attributes which are not mandatory requirements, but has taken the implicit information into account collected directly from the stakeholders.

We follow the perspective of the problem demands priority in this paper and propose a method called Automated Requirement Ranking Approach (RRA), which is based on the following basic characteristics. First, a combined set of priorities are to be extracted from the human decision makers. Second, the RRA is organized according to the iterative scheme that allows deciding when to stop the elicitation process based on measures tradeoffs between inducing effort and accuracy obtained ranks. With reasonable efforts, the method can be applied to as many as 100 requirements.

Third, an alternative method of assigning each request a specific class among a number of different priority classes, such as, in numerical order[6,7] and the first 10 requirements[8]. Well-known approaches ([9-11 etc..]) shows the characterization by the criteria used by each method, and a ranking technique exploits.

The paper is structured as follows: section-2 presents

* *Author for correspondence*

the proposed Automatic Requirement Ranking Approach, and section-3 presents the evaluation and results. Finally, section-4 presents the conclusion.

2. Proposed Automatic Requirement Ranking Approach

In order to illustrate the present application automatically Requirement Ranking Approach (RRA), first it will explain the process priority in terms of machine learning special techniques to give an account to access the intuitive algorithm works on the basis of an example of applying the algorithm which help you to introduce a set of fundamental concepts.

2.1 Approach Overview

We believe the ultimate collection of requirements $Req= \{r_1,r_2,....,r_n\}$ that has to be ranked, and to determine the total space as a set of mandatory pairs $T = \{(r_i, r_j); i < j\}$. We call ordering relationship between the two conditions that can be drawn from the decision maker as a priority.

In a unordered request couple pair of requirements (r_i, r_j) requires the decision-maker has not given priority, i.e., its priority is unknown. We estimate the importance of such costs for the user to run are derived attributes of each application; define the duties of the ranking scale. A range of functions can be linked to these orderings as R_{Att}. We are expected to define the requirements, as shown in Figure 1 which have the rank of the output through the RRA process.

2.2 Requirement Ranking Process

The need for a range of human activities, the obstacles in the process of calculating machine is shown in Figure 2. Input and output data related to the basic curriculum of three steps 1, where they are: the Requirements construction set (*Req*), priorities through decision makers *(P),* the set of Ranking Methods (R_{Att}), encoding the requirement attributes, and the Final Approximated Requirement Rank (*N*) with the consequent process iteration.

The method is based on a set of repetitions of the three steps, which are detailed herein.

- **Pairing of Requirement:** The relative strength of the automated process you need to know to pick from a set of requirements that are set to i.e., disorganized application Couples, as defined in accordance with the rules for sampling. Privacy sampling may be a random choice or it can be taken into account in the ranking calculated in the previous iteration.
- **Extraction of Priority:** It takes a collection of requirements, leading to a pair of input and output, and build the prototype stage based on the priorities expressed in the decision to produce a set of ordered pairs as needed.
- **Learning of Priority:** Due to partial renting priority stakeholders and eventually set Ranking functions, learning algorithm produces approximate unknown desires, and then correspondent approximate order of requirements.

An Approximate Rank, i.e., the output of the process, represents to estimate the correct order, and it can become the input for the next iteration of the process. The results

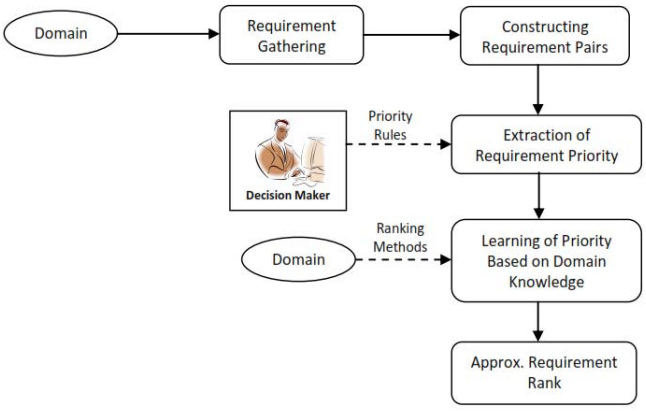

Figure 1. Basic steps of RRA process.

Algorithm-1: Requirement Ranking

Input :

$Req\ [\ ] = \{\ r_1, \ldots, r_z\}$ //-- Sets of Requirements
$R_{Att} = \{\ f_1, \ldots, f_p\}$ //-- Constraints and priority defined based on *Req*
$E_p = \{\ (r_i, r_j);\ i < j \mid E_p\ (r_i, r_j\ ) \neq 0\ \}$ // -- Extracted pair wise preferences

Output :

$N\ (r)\ (N : Req \rightarrow R)$ //-- Ranking function for *Req*

Begin

$W = initialize\ (E_p)$ //-- Uniform weightage of extracted pairs.
$C = maxNumberLearningCycle$ //-- Number of learning cycle.

for $c=1;\ c<=\ C$ ***loop***
$L_c\ (r) = LWC\ (Req,\ E_p\ ,\ R_{Att}\ , W\ );$
// -- Learning most preference pair based on $f \epsilon\ R_{Att}$
// -- *Using a Weak Classifier.*
$\beta_t = ComputeCoefficient\ (L_c\ (r),\ W\ );$
$W_C = ComputeCriticalPair\_Weightage\ (E_p\ , W,\ L_c\ (r));$
End for
$N(r) = \sum_{c=1}^{C} \beta_c L_c(r);$

End

are considered accurate enough to stop the process of learning to step up the frequency of the approximated Rank, *R* as output.

The advantage of the enhanced approach to the transformation of learning priority is described[12,13] for the alignment of the structure. To be precise, in the middle of the range of incentives linear method combining partial orderings will be able to produce an accurate prediction. Algorithm-1 describes the method of requirement ranking as follows.

Requirement Ranking Algorithm performs *C* cycle that calculates a partial order $L_c : Req \rightarrow$ R, called as weak classifier, On the basis of the requirements set *Req*, the set of the extracted Pairs E_p, the Ranking methods R_{Att}, and the set of critical Pairs W_C. In recognition of the previous cycle that will eventually reduce the classification error with respect to user preferences and to calculate the grading method of the symptoms should be considered. It is possible to set a number of requirements in order to be considered part of the weak classifiers, in our experiments we refer to the *L(c)* as described in[14,15]. It is a set of results for each of the requirements of the cycle *c*, and as a result produces a binary classification that defines subsets of the relationship between the priorities.

We compute a value for the parameter β_t. The value of *N*, user preferences, and the choice of tasks $R_{Att,}$ to minimize the error in between. We have to figure out the part of the procedure in order to be pass it on to the next cycle, the count is set for what will be a critical pair L_{c+1}. Consumer preferences and ranking functions R_{Att} in such a way as to reduce the damage in relation to the final rank. What sets an example of how to calculate the user's response is crucial in terms of the values of which have not been classified by the L_C, assigning values to update the distribution of *D*. Assigning values have not been classified with the proper functioning of the L_c with respect to user feedback E_p. This must be ordered in the next cycle of the algorithm correctly when, L_{c+1} is computed.

Let us describe a set of four requirements, that are needed to consider the issue of priority $Req = \{r_1, r_2, r_3, r_4\}$, where set of features and functions that are associated with ranking are given $R_{Att} = \{f_1, f_2\}$. The initial rank, which we consider is defined as follows: $R\_ f_1(r_3) = 1$, $R\_ f_1(r_2) = 2$, $R\_f_1(r_1) = 3$, and $R\_f_1(r_4) = 4$, introduces a priority relation where, $r_4 < r_1 < r_2 < r_3$; in a similar way, the second ranking method is defined as $R\_ f_2(r_1) = 1$, $R\_ f_2(r_3) = 2$, $R\_f_2(r_4) = 3$, and $R\_ f_2(r_2) = 4$, according to the following priority relation $r_2 < r_4 < r_3 < r_1$.

Decision Maker for accomplishing the task of the review is considered a priority for us to say that as, (r_1, r_2) and (r_2, r_4) pairs, generating a priority relationship with

the subsample $E_p = \{ r_2 < r_1 , r_4 < r_2 \}$. Suppose we defined above *Req*, R_{Att}, and E_p, it given the need for input on the ranking algorithm, and simulate the execution.

3. Evaluation

Evaluation method of priorities is not an easy task. Although artificial simulation of the experimental evaluation of the real and the virtual data items are to be carried out to control experiments, however, both methods suffer from certain limitations. Such as requirement rank and limit weaknesses internal assessment approach of the above, we used different settings valuation simulations with synthetic data and case studies with stakeholders.

3.1 Measures

There is disagreement between the two lines of the same size requirements as the base for a quantitative assessment of the relationship. We can get a more comprehensive measure of disagreement calculating the pair wise disagreement universe obligatory pairs. The total disagreement (DA_{Total}) of the two rankings *X* and *Y* is defined as,

$$DA_{Total}(X,Y) = \sum_{T} d_{X,Y}(r_a, r_b) \quad (1)$$

and its normalized disagreement $DA_{Normalization}$ definition is,

$$DA_{Normalization}(X,Y) = \frac{1}{|T|} DA_{Total}(X,Y) \quad (2)$$

Measures disagreement estimate and compare the position of the target and the target is to look at the lines, or apply an attribute ranking. In the first case, the mismatch is preferred to measure the accuracy of the result of the process. Lower value means better range of requirements.

3.2 Results

It is obtained using the following method to set the requirements of the investigation carried out by the artificially generated. Unique identifiers are defined by a number of requirements. Each condition describes a set of attributes, which are integer values ranging from 0 to the cardinality of the set requirements. Attribute values are assigned to each attribute in such a way that induces an overall ranking of a range of needs. Total ranking over the set of requests generated as a reference target position, namely, K, which is in part can be covered with one of the rows of encrypted attributes. This procedure allows the parametric generation data set with a growing number of requests. After the steps in the process of determining the priority rank of the request, as shown in Figure 1, examples of the processes that are simulated using the presented approach.

The first step in the process is a priority associated with the model, initially to provide for all the needs of the target range. For example cardinality of the opening set of z/2 way and is defined as the need for $r_i \in Req$, it is considered a member of at least one pair in the initial set. The second step is, in fact, benefit, not the only step needed to explain human input. We simulate the review of priority as follows:

First, the pair (r_i, r_j) is to select a subset of the Cartesian product $Req \times Req$ is as follows, and we prefer to decision-makers to simulate the extraction with the restriction that $i \neq j$ and $(r_i , r_j) \neq (r_j , r_i)$. The priority task of the E_p (r_i , r_j) simply put directly on the value of the rated operating method. Goal of the simulation is limited by the type of behaviour we retrieved from the monotonous, that is, we can assume that the process of provision of inconsistent responses, particularly artificial decision maker. The third step is required ranking algorithm, *N(r)*, as a result of the fully range *Req* prefer to set the approximate estimated production invokes the appropriate priority.

Experimental Comparison between Requirement ranking and CBRank was conducted on a set of requirements with cardinalities *z* is 20, 40, 60, 80, and 100, respectively. For each set of requirements, run by priority processes, and computed the DA_{Total} and $DA_{Normalization}$ corresponding to a given number of extracted pair preferences.

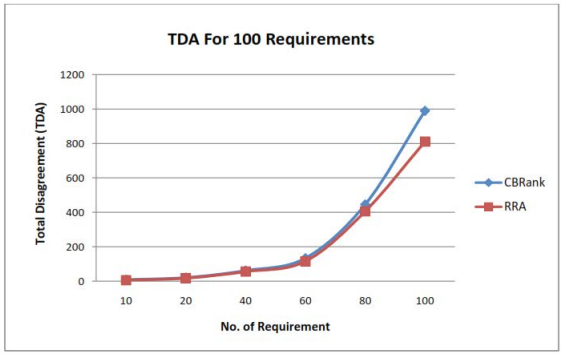

Figure 2. DA_{Total} Comparison between CBRank and RRA.

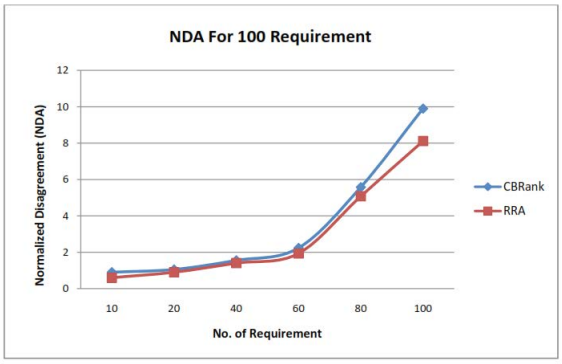

Figure 3. $DA_{Normalization}$ Comparison between CBRank and RRA.

The results are measured with two methods to reduce the differences in growth between the pairs of RRA and CBRank, shows more effective than a low number, especially in the extracted pairs, as shown in the plot in Figure 3. For instance, the difference between the measured differences with the RRA and CBRank in the case of 100 requirements is about 179 pairs, but considerably larger than the difference in the duration of the 10 requirements, which are three pairs. Also, in this case the difference between the number of pairs that are extracted in the RRA and CBRank the normalized differences is 1.79 increases in the case of 100 requirement for CBRank, which conclude the improvisation RRA.

4. Conclusion

In this work we provided, an automated learning technique by taking the advantage of the method which is presented and implemented. A detailed description of the requirement ranking algorithm (RRA) is for the purpose of requirements priority. The requirement ranking method follows a case based scheme to solve the problem, with a new solution, according to previous solutions to similar problem. In the circumstance of the requirements priority, few examples were taken from project stakeholders to pair-up the requirements that assigns priority based on priority rules, and is used to calculate the approximate ranks for the entire set. Experimental measurements were carried out using RRA on various issues of priority, the number of changing requirements and extracted number of pairs for compute ranking. The result shows improvisation and suggesting the requirement priority to be more accurate in future works.

5. References

1. Perini Anna, Susi Angelo and Avesani Paolo. A Machine Learning Approach to Software Requirements Prioritization. IEEE Transactions On Software Engineering. 2013 April; 39(4).
2. Avesani P, Bazzanella C, Perini A and Susi A. Facing Scalability Issues in Requirements Prioritization with Machine Learning Techniques. Proc. 13th IEEE Int'l Conf. Requirements Eng. 2005 Sept; p. 297-306.
3. Peter H, Olson D and Rodgers T. Multi-Criteria Preference Analysis for Systematic Requirements Negotiation. Proc. 26th Int'l Computer Software and Applications Conf. 2002 Aug; p. 887-92.
4. Moisiadis F. Prioritising Software Requirements. Proc. Int'l Conf. Software Eng. Research and Practice. 2002 June.
5. The AN and Ruhe G. Requirements Negotiation under Incompleteness and Uncertainty. Proc. 15th Int'l Conf. Software Eng. and Knowledge Eng. 2003 July; p. 586-93.
6. Akao Y. Quality Function Deployment: Integrating Customer Requirements into Product Design. Productivity Press. 1990.
7. Freund Y, Iyer RD, Schapire RE and Singer Y. An Efficient Boosting Algorithm for Combining Preferences. Proc. 15th Int'l Conf. Machine Learning. 1998; p. 170-78.
8. Hivert F, Novelli J-C and Thibon J-Y. The Algebra of Binary Search Trees. Theoretical Computer Science. 2005; 339(1):129165.
9. Leffingwell D and Widrig D. Addison-Wesley Longman Inc.: Managing Software Requirements: A Unified Approach. 2000.
10. Beck K. Addison-Wesley: Extreme Programming Explained. 1999.
11. Daneva M and Herrmann A. Requirements Prioritization Based on Benefit and Cost Prediction: A Method Classification Framework. Proc. 34th Euromicro Conf. Software Eng. and Advanced Applications. 2008; p. 240-47.
12. Lehtola L. HUT/Department of Computer Science: Providing Value by Prioritizing Requirements Throughout Product Development: State of Practice and Suitability of Prioritization Methods. PhD thesis. 2006.
13. Sivzattian S and Nuseibeh B. Linking the Selection of Requirements to Market Value: A Portfolio-Based Approach. Proc. Seventh Int'l Workshop Requirements Eng.: Foundation for Software Quality. 2001 June.
14. Berander P and Andrews A. Requirements Prioritization. Springer: Eng. and Managing Software Requirements, Aurum A and Wohlin C, eds. 2005.
15. Davis A. Dorset House: Just Enough Requirements Management: Where Software Development Meets Marketing. 2005.
16. Siddiqi JIA and Shekaran MC. Requirements Engineering: The Emerging Wisdom. IEEE Software. 1996 Mar; 13(2) p. 15-19.
17. Karlsson J, Wohlin C and Regnell B. An Evaluation of

Methods for Prioritizing Software Requirements. Information and Software Technology. 1998; 39(14/15) p. 939-47.
18. Wiegers KE. Microsoft Press: Software Requirements. Best Practices. 1999.
19. Lauesen S. Addison Wesley: Software Requirements: Styles and Techniques. 2002.
20. Finkelstein A, Harman M, Mansouri SA, Ren J and Zhang Y. A Search Based Approach to Fairness Analysis in Requirement Assignments to Aid Negotiation, Mediation and Decision Making. Requirements Eng. 2009; 14(4):231-45.
21. Herrmann A and Daneva M. Requirements Prioritization Based on Benefit and Cost Prediction: An Agenda for Future Research. Proc. IEEE 16th Int'l Conf. Requirements Eng. 2008; p. 125-34.
22. Maheshwari V, Prasanna M. Generation of Test Case using Automation in Software Systems - A Review. Indian Journal of Science and Technology. 2015 Dec; 8(35). DOI: 10.17485/ijst/2015/v8i35/72881.
23. Singh Rajesh, Kuchhal Piyush, Gehlot Anita, Choudhury Sushabhan. Design and Implementation of Energy Efficient Home Automation System. Indian Journal of Science and Technology. 2016 Feb; 9(6). DOI: 10.17485/ijst/2016/v9i6/84141.
24. Alshareet OM. An Empirical Study to Develop a Decision Support System (DSS) for Measuring the Impact of Quality Measurements over Agile Software Development (ASD). Indian Journal of Science and Technology. 2015 July; 8(15). DOI: 10.17485/ijst/2015/v8i15/70774.