

Applying MCRDR to a Multidisciplinary Domain

Ivan Bindoff¹, Byeong Ho Kang¹, Tristan Ling¹, Peter Tenni², Prof. Gregory Peterson²

¹ University of Tasmania, School of Computing
{ibindoff, bhkang, trling}@utas.edu.au

² University of Tasmania, Unit for Medical Outcomes and Research Evaluations
{Peter.Tenni, G.Peterson}@utas.edu.au

Abstract. This paper details updated results concerning an implementation of a Multiple Classification Ripple Down Rules (MCRDR) system which can be used to provide quality Decision Support Services to pharmacists practicing medication reviews (MRs), particularly for high risk patients. The system was trained on 126 genuine cases by an expert in the field; over the course of 19 hours the system had learned 268 rules and was considered to encompass over 80% of the domain. Furthermore, the system was found able to improve the quality and consistency of the medication review reports produced, as it was shown that there was a high incidence of missed classifications under normal conditions, which were repaired by the system automatically. However, shortcomings were identified including an inability to handle absent data, and shortcomings concerning standardization in the domain, proposals to solve these shortcomings are included.

1 Introduction

Sub-optimal drug usage is a serious concern both in Australia and overseas [1, 2], resulting in at least 80,000 hospital admissions annually - approximately 12% of all medical admissions - the majority of these concerning elderly patients [3]. MRs are seen as an effective way to improve drug usage. However, the quality of MRs produced is inconsistent across reviewers.

This paper continues discussion commenced in earlier publications by the authors in 2006 and 2007 in which an Intelligent Decision Support System was developed in an attempt to improve the quality of MRs [4, 5]. It was suggested that to improve the consistency and quality of MRs it would be prudent to develop medication management software which includes Intelligent Decision Support features. Prior to this, the majority of incarnations of medication management software for producing MRs has lacked any genuinely “Intelligent” form of Decision Support features [6]. In response to this suggestion, a software system for medication management was developed that utilized the MCRDR method to provide Intelligent Decision Support Services in the multidisciplinary field of MR. [7, 8].

2 Medication Reviews

MR is a burgeoning area in Australia and other countries, with MRs seen to be an effective way of improving drug usage and reducing drug related hospital admissions, particularly in the elderly and other high risk patients [1, 3].

To perform a MR, Pharmacists assess potential Drug Related Problems (DRPs) in a patient by examining various patient records, primarily their medical history, any available pathology results, and their drug regime (past and current) [9].

The expert looks for a variety of indicators between the case details provided checking for known problems, such as an: Untreated Indication – where a patient has a medical condition which requires treatment but doesn't have the treatment; Contributing Drugs – where a patient has a condition and is on a drug which can cause or exacerbate said condition; High Dosage – where a patient is potentially on a too high dosage because of a combination of drugs with similar ingredients; Inappropriate Drug – where a patient is on a drug that is designed to treat a condition they don't seem to have or is contraindicated in their condition; and many others besides. Once these indicators have been identified a statement is produced explaining each problem, or potential problem, and often what the appropriate course of action is.

3 Methodology

In order to produce a medication management system with intelligent decision support features it was necessary to produce two major software elements. The first was a standard implementation of a database “front-end” from which it is possible for a user to enter all the details of a given patient's case, or at least those parts which are relevant to the chosen domain, and which was sufficiently computationally expressive to be machine understandable which shall be referred to after this point simply as the “Domain Model”. The second was an implementation of a Multiple Classification Ripple Down Rules engine which can sufficiently encapsulate the types of conditions and knowledge required for the domain and facilitate the design of an interface from which the engine can be operated, particularly during the Knowledge Acquisition phase.

3.1. Domain Model

The design of the database to store the MR cases was considered to be relatively trivial, and was given relatively little consideration during initial development. The preliminary design was taken from existing medication management software packages, and then modified as required to allow for basic machine readability. The 126 cases considered in this study were then inserted into the database using a simple script which converted them from their current Mediflags [10] format.

3.2. Ripple Down Rules

Ripple Down Rules (RDR) is an approach to building KBSs that allows the user to incrementally build the knowledge base while the system is in use, with no outside assistance or training from a knowledge engineer [8]. It generally follows a forward-chaining rule-based approach to building a KBS. However, it differs from standard rule based systems since new rules are added in the context in which they are suggested.

Observations from attempts at expert system maintenance led to the realisation that the expert often provides justification for why their conclusion is correct, rather than providing the reasoning process they undertook to reach this conclusion. That is, they say 'why' a conclusion is right, rather than 'how'. An example of this would be the expert stating "I know case X has conclusion 1 because they exhibit features r, g and n". Furthermore, experts are seen to be particularly good at providing comparison between two cases and distinguishing the features which are relevant to their different classifications [11]. With these observations in mind an attempt was made at producing a system which mimicked this approach to reasoning, with RDR being the end result.

The resultant RDR structure is that of a binary tree or a decision list [12], with exceptions for rules which are further decision lists. The decision list model is more intuitive since, in practice, the tree would have a fairly shallow depth of correction [13]. The inferencing process works by evaluating each rule in the first list in turn until a rule is satisfied, then evaluating each rule of the decision list returned by that satisfied rule similarly until no further rules are satisfied. The classification that was bound to the last rule that was satisfied is given.

3.3. Multiple Classification Ripple Down Rules

The RDR method described above is limited by its inability to produce multiple conclusions for a case. To allow for this capability - as this domain must - MCRDR should be considered [14] to avoid the exponential growth of the knowledge base that would result were compound classifications to be used.

MCRDR is extremely similar to RDR, preserving the advantages and essential strategy of RDR, but able to return multiple classifications. Contrasting with RDR, MCRDR evaluates all rules in the first level of the knowledge base then evaluates the next level for all rules that were satisfied and so on, maintaining a list of classifications that should fire, until there are no more children to evaluate or none of the rules can be satisfied by the current case [13].

3.3.1. Knowledge Acquisition

Knowledge Acquisition is required when a case has been classified incorrectly or is missing a classification. It is divided into three separate steps: Acquiring new classification (or conclusion), locating the new rule, and acquiring the new rule.

Acquiring the new classification is trivial; the system merely prompts the expert to state it [13]. To acquire the new rule the expert is asked to first select valid conditions from the current case that indicate a given classification. The rule they have created thus far is then compared against the cornerstone case base. If any cornerstone cases would fire on this new rule the expert is asked to select extra condition(s) for the rule from a difference list (see **Table 1**) between the presented case and one of the cornerstone cases. A cornerstone case is a case for which the knowledge had previously been modified and which is valid under the current context [15]. The system then re-tests all cornerstone cases in the list against the appended set of conditions, removing cases from the list that are no longer satisfied. The system repeats this process until there are no remaining cornerstone cases in the list to satisfy the rule [13] or alternatively the expert has stated that the cornerstone cases that remain *should* fire on the new rule, indicating the classification was simply missed on it earlier.

Table 1. Example of a decision list from [8, 16-18]. The list can contain negated conditions.

| <i>Cornerstone case</i> | <i>Current test case</i> | <i>Difference list</i> |
|-------------------------|--------------------------|------------------------|
| Rain | Rain, Meeting | Meeting |
| Meeting | Meeting | Not applicable |
| Hot | | Not(Hot) |

To determine where the new rule must go it must first be determined what type of wrong classification is being made. The three possibilities are listed in **Table 2**.

Table 2. The three ways in which new rules correct a knowledge base [13].

| Wrong Classifications | To correct the Knowledge Base |
|--|---|
| Wrong classification to be stopped | Add <u>stopping</u> rule at the end of the path |
| Wrong classification replaced by correct | Add a rule at the end of the path |
| A new independent classification | Add a rule at the root |

4 Results and Discussion

The system was handed over to the expert with absolutely no knowledge or conclusions pre-loaded. The expert was wholly responsible for populating the knowledge base. Over the course of ~19 hours they were able to add the rules required to correctly classify 126 genuine MR cases that had been pre-loaded into the system.

4.4. Growth of Knowledge Base

It is observed in **Figure 1** that the number of rules in the system progressed linearly as more cases were analysed, at an average rate of 2.04 rules per case. **Figure 1** suggests that the system was still learning heavily until around the 250th rule, at which point the learning rate began to drop off and a plateau began to develop. Previous RDR sys-

tems have been shown to produce a flattening pattern when the knowledge base reaches approximately 80% of domain coverage [13].

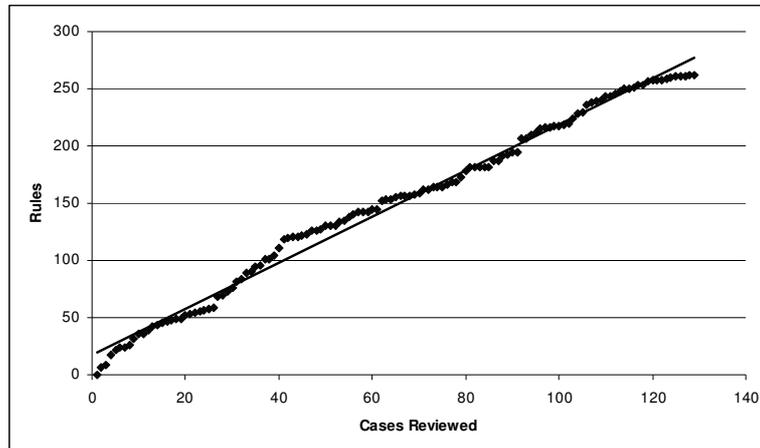


Fig. 1. The number of rules in the system grows linearly, but begins to flatten off.

4.5. Correct Conclusions Found

It was estimated by the expert at cessation of the experiment that the system had encapsulated around 80% of the domain [19], this estimation is supported by the evidence shown in **Figure 2**. It can be seen that the average number of correct classifications the system provided rose quite steadily into the 80th percentile, although the percentage correct from case to case did vary quite a lot, as is to be expected when results are still largely influenced by the early heavy learning phase.

The expert predicted potential classification rates in the order of 90% with this system [19], and it would appear that his estimation was reasonable. Past systems of this nature have demonstrated that despite a flattening pattern commencing, the domain coverage has still continued to grow by an additional 10-15%, although it is conceded that an extremely large number of cases had to be considered for this to be possible [13, 15]. These figures are justified by following the trend-line in **Figure 2** which shows the average of correct conclusions provided by the system for each group of 5 cases analysed, although it is conceded that this trend-line is only a rough approximation. Even following the flattening pattern demonstrated in **Figure 1** it is easy to imagine the system reaching the order of 90% or above, given another 50-150 cases to train with. It should be noted that this is not actually a large amount, when it is considered that it took only ~19 hours of expert time to classify the first 126 cases and it is expected that fewer rules will need to be written for any future cases, given that the system already covers 80% of the domain.

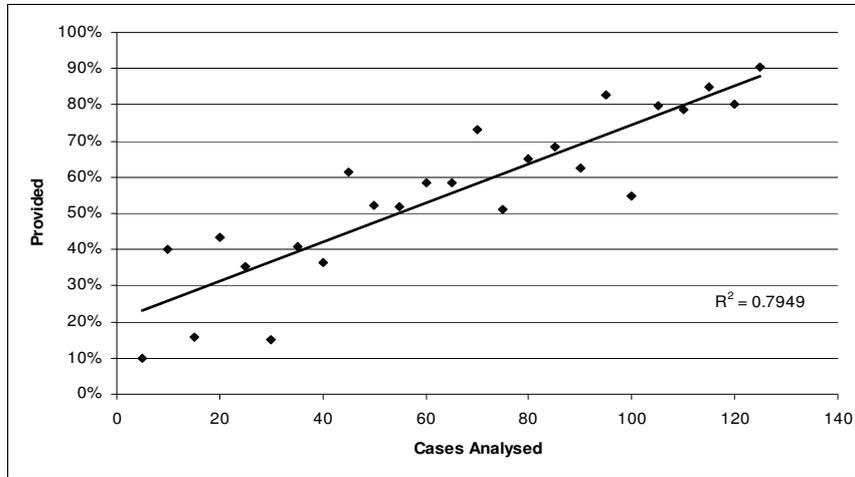


Fig. 2. The grouped average percentage of correct conclusions provided

4.6. Percentage of Classifications Missed

Figure 3 shows that the percentage of cases that received updated classifications reduced dramatically even after only a small number of cases, suggesting the system was rapidly helping to reduce the expert's rate of missed classifications by suggesting the classifications for them. The trend-line is approximate. However, a clearly linear downward progression would be expected without the system assisting the expert.

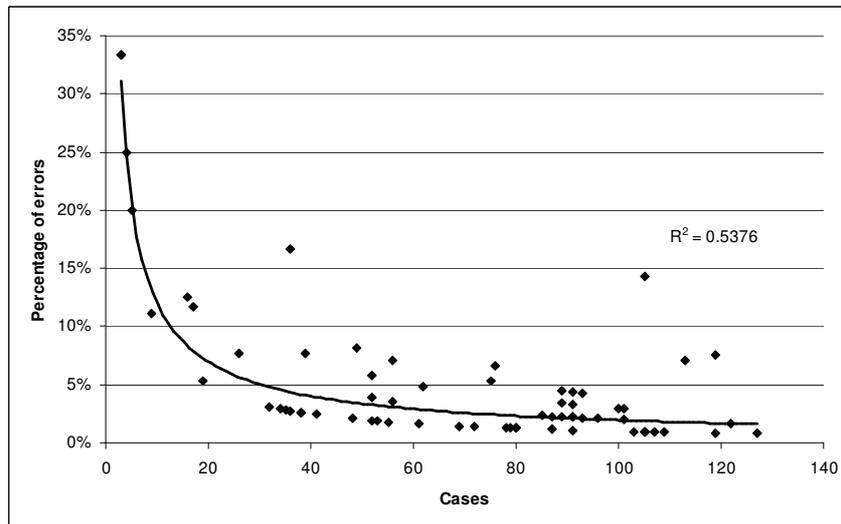


Fig. 3. The percentage of cases that gained new classifications

4.7. Total Errors per Case

It was found that the rate of error in each case was quite high, averaging 13.5% and reaching over 50%. Shown in **Figure 4** is the average errors made in each group of 10 cases considered. The local maxima around case 40 and 100 roughly match periods when the expert's data sources changed.

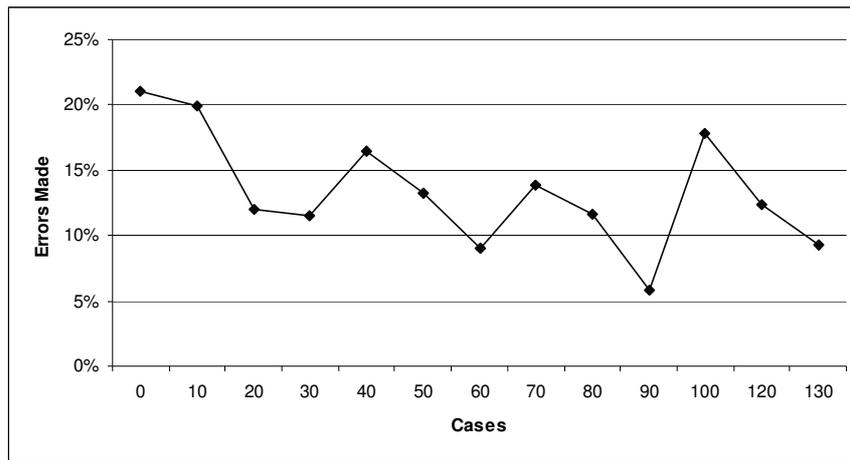


Fig. 4. The final percentage of classifications missed by expert per case

4.8. Maintainability and Usability

It has previously been demonstrated that the complex nature of the multidisciplinary domain of MR did not damage the maintainability and usability of the system [4, 18]. This result did not markedly change through the course of additional testing.

4.8.1. Structure of the Knowledge Base

It can be determined from **Table 3** that the structure of the knowledge base tree was extremely shallow and branchy, suggesting the possibility of an excessive number of exceptions has not eventuated.

Table 3. Structure of the Knowledge Base Tree, 137 branches had a length of 2.

| Tree Property | Value |
|----------------|-------|
| Average Length | 1.79 |
| Length 0 | 41 |
| Length 1 | 86 |
| Length 2 | 137 |
| Length 3 | 25 |
| Length 4 | 5 |
| Length 5 | 5 |

4.9. Observations

The expert was queried regularly during the course of the experimentation, and extensively upon completion. Their comments revealed that although overall satisfied with the power of the system, they were dissatisfied primarily in two areas. Firstly, with the expressability of the domain model, as the expert found themselves unable to create certain desirable rules. For example, it was impossible for the expert to create a rule about a patient's pathology results which expressed the more abstract concept of "increasing" or "decreasing". Similarly, they were challenged with insufficiently detailed levels of grouping for the medications, with the expert finding that grouping on drug types, generic drugs, and commercial drugs too restrictive. Secondly, with fundamental limitations of the MCRDR method, since the expert was unable to create rules which used conclusions of the case as conditions in a new rule. This means the expert is unable to create rules which infer missing attributes of the case, or synonymous attributes, and then create a rule based on these inferred attributes. Further to this, it means the expert cannot necessarily represent the knowledge as efficiently or as faithfully to their own interpretation of it.

5 Conclusions

Initial experimentation suggested that the proposed method using MCRDR could successfully represent knowledge where the knowledge sources (human experts) are inconsistent. The system is shown to have reached about an 80% correct classification rate with less than 20 expert hours and only 126 cases classified – an excellent outcome in the circumstances. The knowledge base structure did not show any major deviations from what would be anticipated in a normal MCRDR system.

From a MR perspective the system was seen to be capable of: providing classifications for a wide range of Drug Related Problems; learning a large portion of the domain of MRs quickly; producing classifications in a timely manner; and importantly, vastly reducing the amount of missed classifications that would otherwise be expected of the reviewer. It is expected that a future incarnation of this system, would be capable of achieving classification rates over 90% [19].

However, it was identified that there are several shortcomings in the current incarnation of the system. Particularly it is known that the domain model is not sufficiently expressive.

Further to this, it was observed that limitations to the method restricted the expert in the creation of genuinely inferred knowledge from the case. This type of functionality is not available in the MCRDR method to date, and to accommodate it will require extensive addition to the method, which is talked about below.

6 Further Work

The current system stands as a satisfactory proof of concept and even in its infancy is considered by the expert to be considerably more powerful than any other medication management system to date. It is clear however that the issues of standardisation and representation of data is still a hurdle, and the only obvious solution is to continue prototyping until a sufficiently expressive domain model is settled upon.

The other issues, concerning the limitations of the method, are still being investigated. It is proposed that to add further conditions based on existing conditions or conclusions of the case it is necessary to add to the underlying tree-like exception based rule structure of the classic MCRDR knowledge base. Imagine a case where the expert wishes to create a rule that reads “IF ConcA & ConcB & Att1 THEN ConcE”. To remain true to the exception based nature of MCRDR this rule must be represented simply as “If Att1 THEN ConcE”, but should be considered only when ConcA and ConcB are already known to be true. To achieve this effect one might add a set of switches to the “If Att1 THEN ConcE” rule. Then, when a requisite conclusion fires it would also turn on its corresponding switch and evaluate that rule. In a situation where all switches on the rule had been activated, and the condition of the rule itself were satisfied, the rule would then fire. This method would result in a graph like structure which maintains every feature of an MCRDR tree, simply extending it with optional extra layers which are dependent on one or more of the above layers’ conclusions. It would even be possible for lower layers to be dependent on higher layers. A possible resultant structure is represented diagrammatically in **Figure 5** below. Further research is currently being undertaken to bring this concept into fruition.

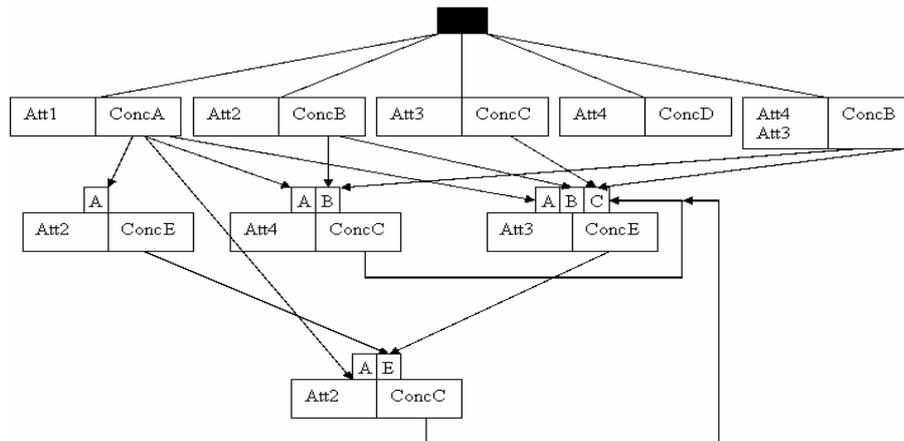


Fig. 5. A graph like structure such as that proposed.

References

1. Peterson, G.:Continuing evidence of inappropriate medication usage in the elderly, in Australian Pharmacist, vol. 23, 2. (2004).
2. Bates, D., Cullen, D., Laird, N., Petersen, L., Small, S., Servi, D., Laffel, G., Sweitzer, B., Shea, B., Hallisey, R.: Incidence of adverse drug events and potential adverse drug events. Implications for prevention. ADE Prevention Study Group. JAMA. vol. 29-34, (1995).
3. Peterson, G.:The future is now: the importance of medication review, in Australian Pharmacist, vol. 268-75. (2002).
4. Bindoff, I., Tenni, P., Kang, B., Peterson, G.:Intelligent Decision Support for Medication Review. In: Advances in Knowledge Acquisition and Management, Conference. Location, (2006).
5. Bindoff, I., Tenni, P., Peterson, G., Kang, B., Jackson, S.: Development of an intelligent decision support system for medication review. J Clin Pharm Ther. vol. 32, 81-8, (2007).
6. Kinrade, W., Review of Domiciliary Medication Management Review Software, Pharmacy Guild of Australia (2003).
7. Aamodt, A., Plaza, E.:Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches, in AICom - Artificial Intelligence Communications, vol. 7, 39-59. (1994).
8. Compton, P., Kang, B., Preston, P., Mulholland, M.:Knowledge Acquisition without Analysis. In: Knowledge Acquisition for Knowledge-Based Systems, Conference. Location, (1993).
9. Tenni, P., Peterson, G., Jackson, S., Hassan, O. to I. Bindoff (2005)
10. MediFlags, <http://www.mediflags.com/>
11. Compton, P., Jansen, R.:A philosophical basis for knowledge acquisition. In: European Knowledge Acquisition for Knowledge-Based Systems, Conference. Location, (1989).
12. Rivest, R.:Learning Decision Lists, in Machine Learning, vol. 2, 229-246. (1987).
13. Kang, B., Compton, P., Preston, P.: Multiple Classification Ripple Down Rules. (1994).
14. Kang, B., Compton, P., Preston, P.:Multiple Classification Ripple Down Rules: Evaluation and Possibilities. In: AIII-Sponsored Banff Knowledge Acquisition for Knowledge-Based Systems, Conference. Location, (1995).
15. Preston, P., Edwards, G., Compton, P.:A 2000 Rule Expert System Without a Knowledge Engineer. In: AIII-Sponsored Banff Knowledge Acquisition for Knowledge-Based Systems, Conference. Location, (1994).
16. Compton, P., Jansen, R.:Cognitive aspects of knowledge acquisition. In: AAAI Spring Consortium, Conference. Location, (1992).
17. Kang, B., Compton, P.: A Maintenance Approach to Case Based Reasoning. (1994).
18. Bindoff, I.:An Intelligent Decision Support System for Medication Review, in Computing, vol. 65. University of Tasmania, Hobart, (2005).
19. Tenni, P. to I. Bindoff (2005)