Title: Rough set based lattice structure for knowledge representation in medical expert systems: low back pain management case study

- **Author Names and Affiliations**

Debarpita Santra\(^a\), Swapan Kumar Basu\(^b\), Jyotsna Kumar Mandal\(^c\), Subrata Goswami\(^d\)

\(^a\)Department of Computer Science and Engineering, Faculty of Engineering, Technology and Management, University of Kalyani, Block C, Nadia, Kalyani, West Bengal, PIN - 741245, India

\(^b\)Department of Computer Science, Institute of Science, Banaras Hindu University, Varanasi-221005, Uttar Pradesh, India

\(^d\)ESI Institute of Pain Management, ESI Hospital Sealdah premises, 301/3 Acharya Prafulla Chandra Road, Kolkata – 700009, West Bengal, India

- **Contact Details of Authors:**

1\(^{st}\) **Author Name:** Debarpita Santra  
**Email Address:** debarpita.cs@gmail.com  
**Telephone Number:** +91-8961468489  
**ORCID of the author:** 0000-0001-5651-5642  
**Full postal address:** Department of Computer Science and Engineering, Faculty of Engineering, Technology and Management, University of Kalyani, Block C, Nadia, Kalyani, West Bengal, PIN - 741245, India

2\(^{nd}\) **Author Name:** Swapan Kumar Basu  
**Email Address:** swapankb@gmail.com  
**Telephone Number:** +91-9450541334  
**Full postal address:** Department of Computer Science, Institute of Science, Banaras Hindu University, Varanasi-221005, Uttar Pradesh, India

3\(^{rd}\) **Author Name:** Jyotsna Kumar Mandal  
**Email Address:** jkm.cse@gmail.com  
**Telephone Number:** +91-9434352214  
**Full postal address:** Department of Computer Science and Engineering, Faculty of Engineering, Technology and Management, University of Kalyani, Block C, Nadia, Kalyani, West Bengal, PIN - 741245, India

4\(^{th}\) **Author Name:** Subrata Goswami  
**Email Address:** drsgoswami@gmail.com  
**Telephone Number:** +91-9830430430  
**Full postal address:** ESI Institute of Pain Management, ESI Hospital Sealdah premises, 301/3 Acharya Prafulla Chandra Road, Kolkata – 700009, West Bengal, India
Abstract

The aim of medical knowledge representation is to capture the detailed domain knowledge in a clinically efficient manner and to offer a reliable resolution with the acquired knowledge. The knowledge base to be used by a medical expert system should allow incremental growth with inclusion of updated knowledge over the time. As knowledge are gathered from a variety of knowledge sources by different knowledge engineers, the problem of redundancy is an important concern here due to increased processing time of knowledge and occupancy of large computational storage to accommodate all the gathered knowledge. Also there may exist many inconsistent knowledge in the knowledge base. In this paper, we have proposed a rough set based lattice structure for knowledge representation in medical expert systems which overcomes the problem of redundancy and inconsistency in knowledge and offers computational efficiency with respect to both time and space. We have also generated an optimal set of decision rules that would be used directly by the inference engine. The reliability of each rule has been measured using a new metric called credibility factor, and the certainty and coverage factors of a decision rule have been re-defined. With a set of decisions rules arranged in descending order according to their reliability measures, the medical expert system will consider the highly reliable and certain rules at first, then it would search for the possible and uncertain rules at later stage, if recommended by physicians. The proposed knowledge representation technique has been illustrated using an example from the domain of low back pain. The proposed scheme ensures completeness, consistency, integrity, non-redundancy, and ease of access.

Keywords: Medical expert system; lattice structure; rough set theory; knowledge-base; low back pain.

1. Introduction

Medical diagnosis is the art of employing physician’s knowledge and intuition in determining the disease of a patient. Exponential growth of medical knowledge and its incredible advancements over years are making the diagnosis process more complicated, thereby deteriorating the quality of medical care. Also, from the speed and reliability point of view, human beings have their limitations, because of excessive patient load, human errors, fatigue, and not having up-to-date knowledge about the recent advancements in medical and associated fields. A medical expert system (Shortliffe (1986); Persidis and Persidis (1991); Abu-Nasser (2017)) which is basically a computer system (devoid of human limitations) powered by AI techniques using the medical domain knowledge, can come to the help and assistance to the doctors.

Efficiency of a medical expert system depends on how much updated knowledge is stored in its knowledge base and how easily it can be retrieved. A knowledge base contains all the relevant facts and rules associated with the problem domain. Knowledge is acquired from different knowledge sources such as domain specific books, related
research papers, patient case histories and wisdom gleaned from expert physicians in related medical domain. In this paper, we have proposed an efficient knowledge representation technique which ensures completeness, consistency, integrity, ease of access, and zero redundancy. The proposed knowledge representation technique automatically generates a set of optimal rules to be used directly by the inference engine of a medical expert system.

The Knowledge Principle: “A system exhibits intelligent understanding and action at a high level of competence primarily because of the specific knowledge that it can bring to bear: the concepts, facts, representations, methods, models, metaphors, and heuristics about its domain of endeavor.” - (Lenat and Feigenbaum (1992))

Design of a knowledge base should allow incremental growth with additions of new medical knowledge over the time. Suppose a knowledge engineer, who is responsible for knowledge acquisition, interviews a domain expert and inserts the acquired knowledge in knowledge base at time $t_1$. At time instant $t_2$ ($t_2 > t_1$) the knowledge engineer gathers additional knowledge from a different knowledge source and also inserts the gathered knowledge in the knowledge base. Some other knowledge engineers may acquire knowledge about the same medical domain from some other knowledge sources. As the knowledge are acquired from a variety of sources like books, journals, research articles, patient case histories etc., the knowledge base may include some redundant and inconsistent knowledge sometimes. To overcome the problem of redundancy in the gathered knowledge, we have proposed a lattice based knowledge representation technique for constructing the knowledge base. As lattice structure (Jongsma (2016)) is purely based on set theory which does not allow duplicate elements, the knowledge base is assured to have zero redundancy. Also for handling inconsistent knowledge in the knowledge base, we have applied Rough Set theory (Pawlak et al. 2008; Pawlak (1998); Zhang et al. 2016; Zhenxiang et al. 1999) on the lattice based design. The Rough Set theory helps us generate certain, possible, and uncertain rules from the inconsistent knowledge. We have also proposed a rule minimization technique for generating optimal rules. This results in a compact knowledge base design.

As we want to acquire sufficient knowledge about a medical domain for a medical expert system in order to help the latter in reliable inferencing, we first have to identify the mutually exclusive knowledge sets for the domain under consideration. More specifically, let us consider $\chi$ as the entire set of knowledge acquired till time $t$. Suppose $\chi_1$, $\chi_2$, $\ldots$, $\chi_k$ $(k > 0)$ are different sets of knowledge, where $\chi_1 \cup \chi_2 \cup \ldots \cup \chi_k = \chi$ and $\chi_1 \cap \chi_2 \cap \ldots \cap \chi_k = \phi$. Every set of different sets of knowledge include the knowledge about radicular low back pain (LBP) (Allegri et al. 2016), knowledge about localized LBP (Natvig et al. 2001), past medical history of patients, knowledge regarding special examinations for LBP (https://stanfordmedicine25.stanford.edu/the25/BackExam.html, accessed on August 19, 2018), and so on. If we take an arbitrary pair $(\chi_i, \chi_j)$ $[i \neq j]$ such that $\chi_i$ will not be used in inference in presence of $\chi_j$, then we say that the pair is mutually exclusive. A knowledge set may also comprise many pairs of mutually exclusive knowledge sets. For example, the knowledge about symptoms of LBP with radicular leg pain and symptoms of LBP without radiculopathy form two different mutually exclusive knowledge sets as they both cannot appear simultaneously during the resolution process.

Designing a knowledge base using a single lattice structure may become large in size. Instead, we represent mutually exclusive knowledge sets using lattice-based structure. If a set of knowledge does not have any mutually exclusive knowledge sets, we would then apply the lattice structure on the whole set. So, each lattice structure would appear as a manageable-sized knowledge module for performing various operations like insertion, deletion or modification of knowledge on it. This kind of modular lattice based approach for knowledge representation will surely enhance efficiency of inference mechanism of the medical expert system. Instead of instantiating all the knowledge modules at a time, this kind of representation would call the modules whenever needed.

The rest of the paper is organized as: section 2 gives an overview of related works. In section 3, we discuss about the basics of lattice and rough set theory. In section 4, we have discussed about our proposed rough set based lattice structure for medical knowledge representation. Section 5 illustrates the proposed knowledge representation technique using a simple example from the domain of LBP. In section 6, we discuss about the basic properties of the knowledge representation. Section 7 concludes the paper.

2. Previous work
Development of an effective knowledge base poses a great challenge to the expert-system researchers. Many knowledge representation techniques (Chakraborty 2008; Karabatak and Ince 2009; Sittig et al. 2010; Tanwar et al. 2010a,b; Arsene et al. 2015) are available in the literature. Among them, production rules are extensively used in medical domain (Kong et al. 2008) and resemble as a good starting point for real-life understanding of the expert systems (Gamberger et al. 2008). Since 1970s, rule-based knowledge representation is being used in many medical expert systems such as MYCIN (Shortliffe et al. 1975; Shortliffe 1976; Shortliffe 2012), PUFF (Aikins et al.1983), UMLS-based CDSS (Achour et al. 2001), CMDS (Huang and Chen (2007)), and so on (Liao (2005); Kuo et al. 2011). Sometimes, production rules have been associated with some probabilistic measures to deal with uncertainties in medical knowledge. Resolution of clinical uncertainties using fuzzy logic (Klir and Yuan (1995); Yen and Langari (1999)) and Bayesian Networks (Friedman et al. 1997; Friedman et al. 1999) is quite popular. Rough Set has also proved its effectiveness in handling inconsistencies in medical knowledge (Tsumoto (1998); Mitra et al. 2006; Ilczuk et al. 2005; Hassnien et al. 2009; Tsumoto (2004)). In this section, we first discuss about knowledge representation techniques in some medical expert systems, and then we discuss some medical applications where rough set has been used. Finally we show how lattice-based structure can be used for knowledge representation.

The production rules have been used for knowledge representation in MYCIN (Shortliffe et al. 1975; Shortliffe (1976); Shortliffe (2012)). Each of the rules which has a premise (Boolean combination of predicate functions) and an action, is associated with a metric called ‘Certainty Factor’ which indicates the strength of a rule. Frame-based knowledge representation has been used by (Langlotz et al. 1988) to describe particular objects or concepts. Using the frames, a decision tree has been constructed for reasoning. Many algebraic relationships exist between the tree - nodes in the decision tree and different medical situations. The qualitative knowledge used for analysis of the expressions is represented using production rules.

The developers of ONCOCIN (Shortliffe et al. 1984) used three kinds of data structures for knowledge representation: control blocks, frames, and rules. While the control blocks were used for encoding procedural knowledge to care for cancer patients, frames were used for storing static protocol knowledge. On the other hand, rules had been used for data-driven or goal-driven inference. ONCOCIN defined a list of contexts for the production rules to be applicable. The modular approach for knowledge representation achieved by developing three different data structures made ONCOCIN more efficient during consultations.

(Weiss et al. 1978) have developed a causal-association network (CASNET) model for capturing the pathophysiological mechanisms of Glaucoma. The knowledge has been represented using three types of data elements corresponding to observations of a patient, states of the causal net, as well as the diagnostic, prognostic, and therapeutic categories. The network produces different types of causal rules with some causal weightages called confidence factors. The rules are logically ordered to produce diagnostic, prognostic and explanatory conclusions. While CASNET model simply propagates the weights among the causal concepts, another approach called ABEL (Patil et al. 1981) is used to symbolically manipulate the weights through various operations. ABEL uses a hierarchical representation of medical knowledge, with the lowest level describing pathophysiological knowledge about diseases, which is aggregated into higher level concepts and relations to ultimately produce syndromic knowledge. This kind of knowledge representation can capture any domain knowledge in different levels of detail.

Case-based reasoning has been used for knowledge representation in many applications (Holt et al. 2005; Bichindaritz et al. 1998; Bichindaritz and Marling (2006); Schmidt et al. 2001) in order to extract knowledge easily from expert physicians in the form of cases. There is also medical application of ontology-driven knowledge representation (Jovic et al. 2007), which holds relationships between concepts such as treatment, clinical procedures, patient data etc. Also production rules can be generated from ontology. (Gamberger et al. 2008) have developed a knowledge base whose descriptive part is designed with ontology and the procedural part has been designed using production rules.

Rough set has been used for knowledge discovery (Polkowski (2013); Huang and Tseng (2004)) in many medical applications. (Own and Abraham (2012)) has proposed a rough set based framework for early intervention and prevention of neurological dysfunction and kernicterus that lead to Egyptian neonatal jaundice. The authors have designed a weighted information table, extracted relevant attributes from the information table, formed a reduced set of weighted attributes, and finally achieved a set of diagnosis rules applicable for neonatal jaundice. A rule-based
A relation $R$ over any set $X$ is a subset of $X \times X$. A relation $R$ is reflexive if for each $x \in X$, $(x, x) \in R$. $R$ is irreflexive if $(x, x) \notin R$. A relation $R$ is symmetric if for all $x, y \in X$, $(x, y) \in R$ implies $(y, x) \in R$. $R$ is antisymmetric if for all $x, y \in X$, $(x, y) \in R$ and $(y, x) \in R$ implies $x = y$. A relation $R$ is asymmetric if for all $x, y \in X$, $(x, y) \in R$ implies $(y, x) \notin R$. A relation $R$ is transitive if for all $x, y, z \in X$, $(x, y) \in R$ and $(y, z) \in R$ implies $(x, z) \in R$. A relation $R$ is an equivalence relation if it is reflexive, symmetric and transitive. Suppose $(x, y) \in R$ and $R$ is an equivalence relation. We denote this phenomenon by $x \equiv R y$. Furthermore, for each $x \in X$, equivalence class of $x$, denoted by $[x]_R$, is defined as the set of all $y \in X$ such that $y \equiv R x$. The set of all such equivalence classes forms a partition of $X$.

A relation $R$ is a partial order if it is reflexive, antisymmetric and transitive. When $R$ is a partial order, we use $x \leq y$ to denote $(x, y) \in R$. The set $X$ together with the partial order is denoted by $(X, \leq)$. We use $P = (X, \leq)$ to denote a partially ordered set or 'poset'. Suppose $Y \subseteq X$. We now define two operators on subsets of the set $X$: ‘meet’ or ‘infimum’ and ‘join’ or ‘supremum’. For any $m \in X$, we say that $m = \text{meet } Y$ iff $\forall y \in Y: m \leq y$, and $\forall m' \in X: (\forall y \in Y: m' \leq y) \Rightarrow m' \leq m$. The first condition says that $m$ is a lower bound of the set $Y$. The second condition says that if $m'$ is another lower bound of $Y$, then it is less than $m$. For this reason, $m$ is also called the greatest lower bound (glb). 

3. Background study

3.1 Preliminaries of lattice theory

Knowledge can also be represented using concept lattice (Zhang et al. 2005; Belohlavek (2008); Aswani Kumar and Srinivasa 2010), which follows lattice structure. There exist some notable works that combine concept lattice and rough set theory (Yang and Xu 2009; Wei and Qi 2010; Yao 2004). Some theories have been proposed for combination of rough set theory and lattice theory (Järvinen 2007; Praba and Mohan 2013; Estaji et al. 2012; Xiao et al., 2014), though the application for the combined theory in medical domain is very limited.

Lattice theory, associated with rough set, can be applied in medical domain in many different ways. In this paper, we have applied rough-lattice theory for generation of exhaustive but optimal set of decision rules. The working principle of concept lattice may not always follow the guidelines of a medical domain, as combinations of different clinical symptoms may indicate a new set of diseases. Pure lattice-based model built on symptoms would be advantageous in this regard. Also, to cope with exponential growth of medical knowledge, an incremental design of knowledge base should be invoked. Generation of decision rules from dynamic databases using incremental rough set approach in less time compared to the other approaches (Blaszczynski and Slowiński 2003; Asharaf et al. 2006; Guo et al. 2005) has been proposed by (Fan et al. 2009).

Our approach uses a lattice of raw knowledge as an information system for rough set, instead of using conventional tabular information system. Though a number of literature exists on rough-lattice theory, its application in medical knowledge representation is rarely found. Compared to the existing approaches, the advantage of using lattice for medical knowledge representation is three-fold: first, the lattice, being a directed graph, is a good way of knowledge representation that shows clearly how every piece of knowledge is related to each other; second, lattice captures an exhaustive set of knowledge in a well-organized and optimized way ensuring completeness and zero-redundancy; third, operations like insertion, deletion, and modification on lattice structure are easy. As knowledge is acquired from different knowledge sources in incremental fashion, overcoming the problem of redundancy is a great achievement as the knowledge base can be stored in optimal space. Therefore the computational time for processing of the stored knowledge would be less.

When rough set uses this lattice-based information system, the generation of optimal diagnostic rules becomes much easier. Overall, the rough set based lattice structure for knowledge representation appears as a computationally efficient technique with respect to both time and space. With the introduction of new reliability metric, the proposed technique is expected to offer more reliability in diagnostic decision making compared to the already-available medical expert systems.
of the set $Y$. It is easy to check that the infimum of $Y$ is unique whenever it exists. However, $m$ is not required to be an element of $Y$. The definition of ‘join’ is similar. For any $s \in X$, we say that $s = \text{join} Y$ iff $\forall y \in Y: y \leq s$, and $\forall s' \in X: (\forall y \in Y: y \leq s') \Rightarrow s \leq s'$. Here, $s$ is also called the least upper bound (lub) of the set $Y$. We denote the lub of $\{a, b\}$ by $(a \land b)$, and lub of $\{a, b\}$ by $(a \lor b)$. The greatest lower bound or the least upper bound may not always exist. A poset $(X, \leq)$ would be called a lattice iff $\forall x, y \in X$, the glb $(x \land y)$ and the lub $(a \lor b)$ always exist.

Finite posets are often depicted graphically using Hasse diagram. To define a Hasse diagram, we first define a relation cover as follows. For any two elements $(x, y) \in X$, $y$ covers $x$ if there is no element $z (\forall z \in X)$ with $x \leq z \leq y$. In this case, $y$ is an upper cover of $x$ and $x$ is a lower cover of $y$. A Hasse diagram of a poset is a graph with the property that there is an edge from $x$ to $y$ iff $y$ covers $x$. Furthermore, while drawing the graph on a Euclidean plane, $x$ is drawn lower than $y$ when $y$ covers $x$. This allows us to suppress the directional arrows in the edges.

3.2 Basics of rough set theory

Rough set theory, proposed by Pawlak in 1982, is a mathematical approach to deal with inconsistency, imprecision, and incompleteness in knowledge. The rough set theory assumes that every object in the universe is associated with some information or knowledge. As an example, we consider three diseases CFJ (Cervical Facet Joint pain) (Manchikanti et al. 2004), discogenic pain (Saal and Saal (2000)), and myofascial pain syndrome (MPS) (Malanga and Colon (2010)) as three objects from the LBP domain. Two symptoms namely "LBP without radiculopathy" and "no LBP at rest" are common in all the three diseases. So, we can say that the three diseases are characterized by these two symptoms. Therefore, these two symptoms are basically the information associated with the three diseases. In this case, as the three diseases are characterized by the same information, these three diseases are called indiscernible in view of the available information (i.e. the two symptoms under consideration) about them.

The basis of rough set theory is an information system. Formally, an information system may be expressed as a pair $S = (U, A)$, where $U$ and $A$ are finite nonempty sets called the ‘universe’ and the set of ‘attributes’, respectively. Every attribute $a \in A$ is associated with a set $V_a$ of its values, called the domain of $a$. Any subset $B$ of $A$ determines a binary relation $I(B)$ on $U$, which will be called an indiscernibility relation. The indiscernibility relation is an equivalence relation. Equivalence classes of the relation $I(B)$ are referred to as $B$-elementary sets or $B$-granules. Any finite union of elementary sets is called a definable set. If a set of attributes and its superset define the same indiscernibility relation (i.e. if elementary sets of both relations are identical), then any attribute that belongs to the superset and not to the set is redundant. A set of attributes with no redundant attribute is called minimal (or independent). The set $P$ of attributes is the reduct (or covering) of another set $Q$ of attributes if $P$ is minimal and the indiscernibility relations defined by $P$ and $Q$ are the same.

The attributes of an information system may further be classified as ‘condition’ and ‘decision’ attributes. So, the previously designed information system is modified as $S = (U, C, D)$, where $C$ and $D$ are disjoint sets of condition and decision attributes respectively. Here, $C \cup D = A$. This kind of information system is called a decision table. If we take an element $x \in U$, we can induce a decision rule as follows: $[c_1(x), c_2(x), \ldots, c_n(x)] \rightarrow [d_1(x), d_2(x), \ldots, d_m(x)]$ or $C \rightarrow D$, where $C = \{c_1, c_2, \ldots, c_n\}$ and $D = \{d_1, d_2, \ldots, d_m\}$. Associated with each decision rule $C \rightarrow D$, there are four quality metrics: support, strength, certainty factor, and coverage factor. Decision rules are often represented in a form of “If condition Then decision”. Thus, any decision table can be transformed in a set of “If condition Then decision” rules, from which a set of optimal rules is generated.

Now, suppose $X \subseteq U$, and $B \subseteq A$. We need to characterize set $X$ in terms of attributes from $B$. If we cannot be able to induce a precise/crisp description of elements of set $X$ using $B$, we have to find out the lower and upper approximations of set $X$. So, generation of lower and upper approximations is required to deal with inconsistency in knowledge. The $B$-Lower Approximation of $X$ (denoted by $B^-(X)$) and $B$-Upper Approximation of $X$ (denoted by $B^+(X)$) are formally defined as follows: $B^-(X) = \bigcup_{x \in X} \{B(x): B(x) \subseteq X\}$ and $B^+(X) = \bigcup_{x \in X} \{B(x): B(x) \cap X = \emptyset\}$. The set containing elements from the upper approximation of a concept that are not members of the lower approximation of the concept, is called a boundary region. The set $BN(X)[= \{B^+(X) - B^-(X)\}]$ is called the $B$-Boundary Region of $X$ and thus consists of those objects that we cannot decisively classify into $X$ on the basis of knowledge in $B$. If $BN(X) = \emptyset$, then $X$ is Crisp (exact) w.r.t. $B$. If $BN(X) \neq \emptyset$, then $X$ is Rough (inexact) w.r.t. $B$. A Rough Set may be defined as a set having non-empty boundary regions.
4. Proposed knowledge representation technique

We conceptualize the knowledge as a collection of rules, with each rule containing a condition part \( C \) and a decision part \( D \). \( C \) consists of \( n > 0 \) facts and \( D \) consists of \( m > 0 \) decisions. We denote an attribute-value pair as a fact, and a disease-value pair as a decision. In this work, condition parts hold sets of symptoms and decision parts include sets of diseases. Mathematically we can express a rule \( R \) as follows:

\[
R: C \rightarrow D, \text{ where } C = \prod_{k=1}^{n} f_k \text{ and } D = \sum_{i=1}^{m} D_i, \text{ where } f_k \text{ is the } k^{th} \text{ fact, } D_i \text{ is the } i^{th} \text{ decision.}
\]

Here \( f_k = (\text{Attribute}_k, \text{Value}) \) and \( D_i = (d_i, \text{Value}) \). \( V_d \) is a ternary variable which may accept 3 types of values: \textit{true} (1), \textit{false} (0), or \textit{inconclusive} (2). When \( V_d = 1 \) for \( d_i \), we say that the disease surely occurs for a particular set of symptoms. If \( V_d = 0 \) for \( d_i \), we can conclude that the disease never occurs for a particular set of symptoms. Again, if \( V_d = 2 \) for \( d_i \), we would say that for a particular set of symptoms, \( d_i \) may or may occur. A rule which contains a single fact is called an atomic rule. A rule which contains more than one fact is called a composite rule. We design the knowledge base (KB) as a set of such kinds of rules.

We now build a lattice structure for capturing the unrefined rules gathered directly from the knowledge sources. The lattice of such rules would act as the information system for rough set. The rough set will then remove the inconsistent knowledge hidden in the lattice and would produce a set of optimal decision rules. Figure 1 depicts the proposed technique.

![Figure 1. Rough set based lattice structure for medical knowledge representation](image)

4.1 Lattice structure for knowledge representation

Initially KB is empty i.e. KB = \( \phi \). Every time we encounter a new rule, we add it to the existing KB using ‘set union’ operation. We first insert the atomic rules in KB. Suppose we have \( n > 0 \) atomic rules. So, the knowledge base is modified as KB = KB ∪ \{ \( R | R \) is an atomic rule with fact \( f_i \) where \( 1 \leq i \leq n \) \}. Then we insert composite rules in KB. For this, we combine the facts of the atomic rules in the following way: first we construct the total number of possible groups that can be formed by taking two different facts at a time out of \( n \) distinct facts. Obviously the total number of such groups is \(^nC_2\). So, from \( n \) atomic rules, we can generate at most \(^nC_2\) distinct composite rules whose condition parts would contain exactly two facts. These rules are included in KB. Now we form groups of facts taking three different facts at a time from the set of \( n \) facts and generate composite rules with three facts accordingly. The new knowledge is inserted into the KB. This process continues until we reach to a state where only one group is formed taking \( n \) distinct facts at a time. So, at this stage, only one composite rule with \( n \) facts would be generated. After the construction process is complete, the KB would hold \( (2^n - 1) \) rules. If we consider only the sets of facts in KB, we obtain a set \( S \) which includes \( 2^n \) combinations of facts including \( \phi \). So, the set \( S = \{ \phi, \{f_1\}, \{f_2\}, \{f_3\}, \ldots, \{f_n\}, \{f_1, f_2\}, \{f_1, f_3\}, \ldots, \{f_{n-1}, f_n\}, \{f_1, f_2, f_3\}, \ldots, \{f_{n-2}, f_{n-1}, f_n\}, \ldots, \{f_1, f_2, \ldots, f_{n-1}, f_n\} \} \).

Now we consider a relation \( R \) as subset equality i.e. \( \subseteq \). Suppose \( A, B, C \subseteq S \). We say that \( [S; \subseteq] \) is a poset as
Now, we take two elements $x \in S$ and $y \in S$ and construct $\{x, y\}$. This set would always have a $glb$ as $(x \cap y) \in S$ and $lub$ as $(x \cup y) \in S$. This is true for any $x \in S$ and $y \in S$. So, $S$ is a lattice of order $n$.

We now try to define the conditional weightage of a fact mathematically. Suppose there is a rule which includes $n(> 0)$ facts with priorities $p(\leq n)$ for a particular disease $d$. A fact $f_i$ has priority $p_i$. The priority of each fact is obtained from different knowledge sources. We define the conditional weightage $C_{w_i}$ associated with each fact $f_i$ ($1 \leq i \leq n$) for disease $d$ as $C_{w_i} = p_i/n$.

Now we define a quantifying factor for a rule keeping in mind the uncertainties in a medical domain. A rule can be formally represented using 8-tuple $<C, C_n, d, T^1, T^2, T^3, V_d, CF>$, where $C = \{f_i | f_i$ is a fact and $1 \leq i \leq n\}; C_n = \{C_{w_i} | C_{w_i}$ is the conditional weightage of $f_i\}; d$ is the disease for the pair $<C, C_n>$. $T^3$ denotes a value to measure
the acceptability of the knowledge sources that give evidence about the certain presence of disease d for \( <C, C_o> \); \( T_i^2 \) denotes a value to measure the acceptability of knowledge sources that provide evidence in support of certain absence of disease d for the same pair; \( T_i^3 \) denotes a value to measure the acceptability of knowledge sources which tell us that the disease d may or may not be present in the current context. Each of the values would be between 0 and 1; \( V_i \) is the truth value for d which is decided based on \( T_i^1, T_i^2, T_i^3 \); CF (Credibility Factor) gives some measure of the rule for being believable or trustworthy. One important thing to mention here is that if \( R: C \rightarrow D \) is a rule where \( D = \{ <d_k, V_k> | <d_k, V_k> \text{ is a decision and } 1 \leq k \leq K \} \), we break down R into K sub-rules of the form \( R_k: C \rightarrow <d_k, V_d> \). Each sub-rule also follows the 8-tuple representation.

Acceptability of a knowledge source is measured based on the nature of the source. According to opinions of different experts from the domain of LBP, the various sources of knowledge have been assigned weightages as per their acceptability levels; e.g. national guidelines have been graded as level 1 (highest priority), randomized controlled trials (RCT) (Stolberg et al. 2004) is graded as level 2, existing literature (journals/ articles/ books etc.) belong to level 3 category, patient case studies belong to level 4 category, and recommendations of expert physicians belong to level 5 category (lowest priority). As this kind of gradation is not standardized, we say that, in general, knowledge resources can be classified into q different categories, where level 1 means highest priority and level q is considered as lowest priority.

The generalized formula for calculating \( T_v^m \) (where \( 1 \leq m \leq 3 \)) is as follows: we consider that there are \( K_m \) knowledge sources corresponding to \( T_v^m \). We assume that all the knowledge sources of set \( K_m \) are not of same category, with a set of \( |K_m| \) \( (K_m \subseteq K_m) \) sources belonging to level \( j \) category, where \( 1 \leq j \leq q \). The consideration is that \( K_m \cap K_m \cap \ldots \cap K_m = \phi \) and \( K_m \cup K_m \cup \ldots \cup K_m = K_m \). The generalized forms of \( T_v^1, T_v^2, \text{ and } T_v^3 \) are given in equation (i).

\[
T_v^1 = \frac{x}{w}, \ T_v^2 = \frac{y}{w}, \ T_v^3 = \frac{z}{w} \quad \text{............ (i)}
\]

where
\[
x = (q|K_{11}| + (q - 1)|K_{12}| + (q - 3)|K_{13}| + \ldots + |K_{1(q-1)}| + |K_{1q}|) = (q|K_{21}| + (q - 1)|K_{22}| + (q - 3)|K_{23}| + \ldots + |K_{2(q-1)}| + |K_{2q}|)
\]
\[
y = (q|K_{31}| + (q - 1)|K_{32}| + (q - 3)|K_{33}| + \ldots + |K_{3(q-1)}| + |K_{3q}|)
\]
\[
z = (q|K_{41}| + (q - 1)|K_{42}| + (q - 3)|K_{43}| + \ldots + |K_{4(q-1)}| + |K_{4q}|)
\]
\[
w = (x + y + z)
\]

The implicit assumption is that \( W \neq 0 \). The complemented forms of the parameters are as follows:

\( \sim T_v^1 = 1 - T_v^1 \), which reflects how much unacceptability is there to claim certain presence of disease d for C;

\( \sim T_v^2 = 1 - T_v^2 \), which reflects how much unacceptability is there to claim certain absence of disease d for C;

\( \sim T_v^3 = 1 - T_v^3 \), which reflects how much unacceptability is there to say that disease d may or may not be present.

As we have just seen that a decision is taken based on conditions with different conditional weightages, measuring whether the rule is trustworthy requires careful investigation about different parameters like \( T_v^1, \ K_m, \ldots, K_m \). We now construct a \((3 \times q)\) matrix \( M \) where, the first row is for \( K_1 \), second row is for \( K_2 \), and third row is for \( K_3 \). The columns 1 to q represent different categories of knowledge sources. Our assumption is that indices of \( M \) start from location (1,1). \( M(x,y) = 1 \) if for row \( x \) (\( 1 \leq x \leq 3 \)), there are knowledge sources under category y (\( 1 \leq y \leq q \)), else \( M(x,y) = 0 \). Using \( M \), we determine \( V_d \) of d and also calculate the corresponding CF.

If 1 is present at one or more than one column of only one row \( x \) of \( M \), then \( V_d = 1 \) and \( CF = T_v^1 \) if \( x = 1 \); \( V_d = 0 \) and \( CF = T_v^2 \) if \( x = 2 \); \( V_d = 2 \) and \( CF = T_v^3 \) if \( x = 3 \). Now, consider that 1 is present in multiple rows of \( M \), then we proceed column-wise from left to right checking for first \( y \) (\( \geq 1 \)) when \( M(x,y) = 1 \). So, obviously \( M(x_1, y_1) = 0 \) where \( 1 \leq y_1 < y \) and \( 1 \leq x_1 \leq 3 \). We shall now discuss about two cases: i) \( M(x, y) = 1 \) for a single row \( x \) of \( M \) and for each \( x' \) representing the rest two rows, \( M(x', y) = 0 \); ii) \( M(x, y) = 1 \) for each \( x \) from more than one row.

The first case is very straightforward. In this case, \( V_d = 1 \) and \( CF = T_v^1 \) if \( x = 1 \); \( V_d = 0 \) and \( CF = T_v^2 \) if \( x = 2 \); \( V_d = 2 \) and \( CF = T_v^3 \) if \( x = 3 \). The second case requires considering situations when two rows have value 1 under column y and also when three rows of \( M \) have value 1 at column y. So, there are basically four such combinations as given below:

a) Case: \( M(1,y) = 1, M(2,y) = 1, \) and \( M(3,y) = 0 \)
If \( T_{v}^1 > T_{v}^2 \), then \( V_d = 1 \) and \( CF = T_{v}^1 \). Else if \( T_{v}^1 < T_{v}^2 \), then \( V_d = 0 \) and \( CF = T_{v}^2 \). But if \( T_{v}^1 = T_{v}^2 \), then starting from column \((y+1)\), we would find the first \( y' \) ( \( y < y' \leq q \)) where either \( M(1, y') = 1 \) or \( M(2, y') = 1 \), but not both. If \( M(1, y') = 1 \), then \( V_d = 1 \) and \( CF = T_{v}^1 \); else \( V_d = 0 \) and \( CF = T_{v}^2 \). If no such \( y' \) is found in \( M \), \( V_d = 2 \) and \( CF = T_{v}^3 \).

b) Case: \( M(1, y) = 1 \), \( M(2, y) = 0 \), and \( M(3, y) = 1 \)

If \( T_{v}^1 > T_{v}^3 \), then \( V_d = 1 \) and \( CF = T_{v}^1 \); Else if \( T_{v}^1 < T_{v}^3 \), then \( V_d = 2 \) and \( CF = T_{v}^3 \). But if \( T_{v}^1 = T_{v}^3 \), then starting from column \((y+1)\), we find the first \( y' \) ( \( y < y' \leq q \)) where either \( M(1, y') = 1 \) or \( M(3, y') = 1 \), but not both. If \( M(1, y') = 1 \), then \( V_d = 1 \) and \( CF = T_{v}^1 \); else \( V_d = 2 \) and \( CF = T_{v}^3 \). If no such \( y' \) is found in \( M \), \( V_d = 2 \) and \( CF = T_{v}^3 \).

c) Case: \( M(1, y) = 0 \), \( M(2, y) = 1 \) and \( M(3, y) = 1 \)

If \( T_{v}^2 > T_{v}^3 \), then \( V_d = 0 \) and \( CF = T_{v}^2 \); Else if \( T_{v}^2 < T_{v}^3 \), then \( V_d = 2 \) and \( CF = T_{v}^3 \). But if \( T_{v}^2 = T_{v}^3 \), then starting from column \((y+1)\), we find the first \( y' \) ( \( y < y' \leq q \)) where either \( M(2, y') = 1 \) or \( M(3, y') = 1 \), but not both. If \( M(2, y') = 1 \), then \( V_d = 0 \) and \( CF = T_{v}^2 \); else \( V_d = 2 \) and \( CF = T_{v}^3 \). If no such \( y' \) is found in \( M \), \( V_d = 2 \) and \( CF = T_{v}^3 \).

d) Case: \( M(1, y) = 1 \), \( M(2, y) = 1 \) and \( M(3, y) = 1 \)

If \( T_{v}^1 > T_{v}^2 \) and \( T_{v}^1 > T_{v}^3 \) then \( V_d = 1 \) and \( CF = T_{v}^1 \). Else if \( T_{v}^1 < T_{v}^2 \) and \( T_{v}^2 > T_{v}^3 \), then \( V_d = 0 \) and \( CF = T_{v}^2 \). If \( T_{v}^1 > T_{v}^3 \) and \( T_{v}^3 > T_{v}^2 \), then \( V_d = 2 \) and \( CF = T_{v}^3 \). For other kinds of relationships among \( T_{v}^1, T_{v}^2, T_{v}^3 \) except any equality relation, we find out the maximum of all three and determine \( V_d \) and \( CF \) accordingly. Now, consider that equality relation exists between \( T_{v}^1 \) and \( T_{v}^2 \) and \( T_{v}^2 < T_{v}^3 \). In this case \( V_d = 2 \) and \( CF = T_{v}^3 \). Similar strategy would be followed if \( T_{v}^1 = T_{v}^3 \) and \( T_{v}^1 < T_{v}^2 \), or \( T_{v}^3 = T_{v}^2 \) and \( T_{v}^2 < T_{v}^3 \). For the reverse case, when \( T_{v}^1 = T_{v}^2 \) and \( T_{v}^1 > T_{v}^3 \), we say \( V_d = 2 \) and \( CF = T_{v}^3 \). For the rest cases, \( V_d = 2 \) and \( CF = T_{v}^3 \).

If \( T_{v}^3 = T_{v}^2 = T_{v}^3 \), then starting from column \((y+1)\), we find the first \( y' \) ( \( y < y' \leq q \)) where either \( M(1, y') = 1 \) or \( M(2, y') = 1 \) or \( M(3, y') = 1 \), but not all, provided that no two rows have 1s under column \( y'' \) ( \( y < y'' < y' \)). If \( M(1, y') = 1 \), then \( V_d = 1 \) and \( CF = T_{v}^1 \); if \( M(2, y') = 1 \), then \( V_d = 0 \) and \( CF = T_{v}^2 \); else \( V_d = 2 \) and \( CF = T_{v}^3 \). If no such \( y'' \) and \( y' \) are found in \( M \), \( V_d = 2 \) and \( CF = T_{v}^3 \). But if we have \( M(1, y'') = 1 \) and \( M(2, y'') = 1 \), then we search for the next column \( y'' \) ( \( y'' < y'' < y'' \)), where \( M(x', y'') = 1 \) with \( x' \) being the first or second row. If \( M(1, y'') = 1 \), then \( V_d = 1 \) and \( CF = T_{v}^1 \). If \( M(2, y'') = 1 \), then \( V_d = 0 \) and \( CF = T_{v}^2 \). If no such \( y'' \) is found in \( M \), \( V_d = 2 \) and \( CF = T_{v}^3 \). Likewise, for \( M(2, y') = 1 \) and \( M(3, y') = 1 \), if \( M(2, y') = 1 \), then \( V_d = 0 \) and \( CF = T_{v}^2 \); else \( V_d = 2 \) and \( CF = T_{v}^3 \). If no such \( y' \) is found in \( M \), \( V_d = 2 \) and \( CF = T_{v}^3 \).

Now let us consider two atomic rules \( R_i \): \( C_i \rightarrow D_i \) and \( R_j \): \( C_j \rightarrow D_j \), where \( i \neq j \). Suppose \( D_i = (d, V_d') \) and \( D_j = (d, V_d'') \). From these two rules, a new rule \( R_{ij} \): \( C_i \rightarrow C_j \rightarrow D_j \) is induced, where the conditional weightage for fact \( f_i \) is \( C_{ij}^i \) and that of fact \( f_j \) is \( C_{ij}^j \). The conditional weightages are supplied by knowledge sources. If there is no knowledge source available for \( R_{ij} \), then we would assume that the conditional weightages are same for each fact. As per our consideration, \( D_i \) would also include \( d \) as the disease. If knowledge sources are available for \( R_{ij} \), we consider that \( K_{ij}^1 \) is the set of knowledge sources to denote certain presence of disease \( d \) for \((C_i \cup C_j) \). \( K_{ij}^2 \) is the set of knowledge sources to denote certain absence of disease \( d \), and \( K_{ij}^3 \) is the set of knowledge sources which indicate that disease \( d \) may or may not be present for \((C_i \cup C_j) \).

We now calculate \( T_{m}^{m} [1 \leq m \leq 3] \) for \( R_{ij} \), denoted by \( [T_{m}^{m}]_{ij} \), as per equation (i). Note that, \( [T_{m}^{m}]_{ij} \) is obtained from external knowledge sources for \( R_{ij} \) and it is not the actual value of \( T_{m}^{m} \). If no such knowledge source is available, \( [T_{m}^{m}]_{ij} \) would be 0. The actual value of \( T_{m}^{m} \) for \( R_{ij} \), denoted by \( [T_{m}^{m}]_{ij} \), is obtained as \(((T_{m}^{m})_{ij} + [T_{m}^{m}]_{ij} + [T_{m}^{m}]_{ij})/3 \), where \( [T_{m}^{m}]_{ij} \) and \( [T_{m}^{m}]_{ij} \) represent the values of \( T_{m}^{m} \) for \( R_{i} \) and \( R_{j} \) respectively. If \( [T_{m}^{m}]_{ij} = 0 \), then \( [T_{m}^{m}]_{ij} = ([T_{m}^{m}]_{ij} + [T_{m}^{m}]_{ij})/2 \).
Now we determine the \( V_d \) of \( d \) and \( CF \) of \( R_j \) based on the available knowledge of \( R_i \) and \( R_j \) in the lattice-based KB. The values obtained from the implicit lattice knowledge are termed as \( [V_d]_i \) and \( CF^*(R_0) \). If no knowledge sources are available for \( R_j \), the final \( V_d \) of \( d \) in \( R_j \) is \( [V_d]_j = [V_d]_i \) and the final \( CF \) of \( R_j \) \( CF(R_j) = CF^*(R_j) \).

Case 1: Two constituent rules having the same \( V_d \) for \( d \)

Suppose two rules \( R_i \) and \( R_j \) hold the same truth value for disease \( d \). We say that the newly induced rule \( R_j \) would also have the same truth value. The motivation behind this consideration is that, if father and mother carry the same genetic characteristic, the child also would inherit the same genetic characteristic. We now calculate the credibility factor for \( R_j \) against the decision \( D_j \) as shown in equation (ii). Here, we use a threshold \( \alpha (0 \leq \alpha \leq 1) \) if we want to restrict use of the credibility factor of a rule in some further calculations. The value of \( \alpha \) is application-specific.

\[
CF^*(R_{ij}) = \begin{cases} 
(CF(R_i)C^l_w + CF(R_j)C^l_w), & CF(R_i)C^l_w > \alpha \text{ and } CF(R_j)C^l_w > \alpha \\
CF(R_i)C^l_w, & CF(R_i)C^l_w \leq \alpha \text{ and } CF(R_j)C^l_w > \alpha \\
CF(R_j)C^l_w, & CF(R_i)C^l_w \leq \alpha \text{ and } CF(R_j)C^l_w > \alpha \\
0, & CF(R_i)C^l_w \leq \alpha \text{ and } CF(R_j)C^l_w \leq \alpha 
\end{cases} \tag{ii}
\]

Case 2: Two constituent rules having different \( V_d \) for \( d \)

Two constituent rules \( R_i \) and \( R_j \) may hold different values for \( d \) in three different ways: i) \( (V_d = 1, V_d = 0) \) or vice-versa; ii) \( (V_d = 1, V_d = 2) \) or vice-versa; iii) \( (V_d = 0, V_d = 2) \) or vice-versa. To determine \( [V_d]_d \) for \( d \) in the newly induced rule \( R_j \), we have to consider carefully \( CF(R_i) \) and \( CF(R_j) \). If \( CF(R_i) > CF(R_j) \), then \( [V_d]_d = V_d \). If \( CF(R_i) > CF(R_j) \), then \( [V_d]_d = V_d \). On the other hand, if \( CF(R_i) = CF(R_j) \), then \( [V_d]_d = 2 \). We now determine in equation (iii) how to calculate \( CF^*(R_{ij}) \).

\[
CF^*(R_{ij}) = \begin{cases} 
([CF(R_i)C^l_w - CF(R_j)C^l_w]), & CF(R_i)C^l_w > \alpha \text{ and } CF(R_j)C^l_w > \alpha \\
CF(R_i)C^l_w, & CF(R_i)C^l_w \leq \alpha \text{ and } CF(R_j)C^l_w > \alpha \\
CF(R_j)C^l_w, & CF(R_i)C^l_w \leq \alpha \text{ and } CF(R_j)C^l_w > \alpha \\
0, & CF(R_i)C^l_w \leq \alpha \text{ and } CF(R_j)C^l_w \leq \alpha 
\end{cases} \tag{iii}
\]

Now we describe the mechanism for calculating the credibility factor for the newly induced rule \( R_j \) when we have knowledge sources for \( R_j \). In this scenario, if \( [V_d]_d \) for \( d \) in \( D_j \), as measured from the constituent rules, is same as the truth value as given by the knowledge sources, then we calculate \( \text{exp1} = CF^*(R_{ij}) + CF^*(R_{ij}) \) where \( CF^*(R_{ij}) \) is the credibility factor for \( R_j \) as calculated from supplying knowledge sources. If \( \text{exp1} > 1 \), then \( CF^*(R_{ij}) = 1 \), else \( CF^*(R_{ij}) = \text{exp1} \). But, if \( [V_d]_d \) for \( d \) in \( D_j \) as measured from constituent rules is different from truth value \( ([V_d]_d) \) for \( d \) in \( D_j \), as given by the external evidence, we determine \( [V_d]_d \) and the credibility factor for \( R_j \) as per the following algorithm.

Procedure Credibility_\_R_k\_External\((CF^*(R_0), CF^*(R_j))\)

If \( (CF^*(R_0) > CF^*(R_j)) \) Then
\[
[V_d]_d = [V_d]_j; \\
CF(R_j) = CF^*(R_0) - CF^*(R_j); \\
\text{Elseif} \ (CF^*(R_0) < CF^*(R_j)) \text{ Then} \\
[V_d]_d = [V_d]_j; \\
CF(R_j) = CF^*(R_0) - CF^*(R_j); \\
\text{Else} \\
[V_d]_d = 2; \\
CF(R_j) = [T_3]; \\
\endend
\]

End Credibility_\_R_k\_External

The algorithm takes constant execution time.
So far we have considered only two constituent atomic rules for inducing a new rule. But as per our lattice based KB, a rule \( R_{new} \) at level \( i \) \((i>2)\) may be constructed using a set of \( i \) constituent rules \( R_c, R_{c1}, \ldots, R_{ci} \) \((c>0)\). Suppose all the constituent rules contain \( d \) as part of their decisions. We will first show how to determine \( V_d \) for \( d \) in \( R_{new} \) using the following steps: starting from extreme left of the lattice diagram, we first take two constituent rules \( R_c \) and \( R_{c1} \) from the set of \( i \) rules to construct a dummy rule \( R_{dummy} \). We then check whether \( CF(R_c) \geq CF(R_{c1}) \). If it is, then \( V_d \) for \( R_{dummy} \) will be same as that of \( R_c \). If \( CF(R_c) < CF(R_{c1}) \), then \( V_d \) for \( R_{dummy} \) will be same as that of \( R_{c1} \). If \( CF(R_c) = CF(R_{c1}) \), then \( V_d \) for \( R_{dummy} \) will be same as that of either \( R_c \) or \( R_{c1} \). Now, we take \( R_{dummy} \) and the third constituent rule \( R_{c2} \) to construct another dummy rule \( R_{dummy2} \) and carry out the same procedure as \( R_{dummy1} \) for calculating \( V_d \) of \( d \) for \( R_{dummy2} \). In this way we proceed until we obtain the rule \( R_{dummy(i-1)} \) and calculate its \( V_d \). Now we use the credibility factors of \( R_{dummy(i-1)} \) and that of the \( i^{th} \) constituent rule \( R_{c(i)} \) to calculate the final \( V_d \) of the rule \( R_{new} \).

We now show how to determine the credibility factor for \( d \) in \( R_{new} \). We assume that the condition part of \( R_{new} \) consists of facts \( f_1, f_{k+1}, f_{k+2}, \ldots, f_{k+i} \) \((k>0)\) associated with conditional weightages \( C_{w1}, C_{w2}, \ldots, C_{wi+i} \) respectively. If we consider all the condition parts of constituent rules, then we would find each of the facts \((i-1)\) times. Using the implicit lattice knowledge, we calculate the credibility factor for \( R_{new} \). For this, we consider that a sub-set of constituent rules \( S_i \subset \{R_c, R_{c1}, \ldots, R_{ci}\} \) that hold fact \( f_k \) in the condition parts. Similarly, \( S_{k+1} \subset \{R_c, R_{c1}, \ldots, R_{ci}\} \) holds fact \( f_{k+1} \) and so on. Obviously, \( S_i \cap S_{i+1} \cap \ldots \cap S_{i+1} \neq \phi \). Now consider \( S_i \). We check \( V_d \) of each of rules belonging to \( S_i \). Suppose there are \( m_i \) rules whose \( V_d \) match with that of \( R_{new} \), and \( n_k = \|S_i\| - m_i \) rules whose \( V_d \) do not match with that of \( R_{new} \). We define \( CF_{k+1} = \sum_{a=0}^{m_{k+1}} CF(S^a_k) - \sum_{b=0}^{n_k} CF(S^b_k) \), where \( S^a_k \) \((0 \leq a \leq m_k)\) represents a rule belonging to \( S_i \) with same \( V_d \) as \( R_{new} \), and \( S^b_k \) \((0 \leq b \leq n_k)\) represents a rule belonging to \( S_i \) with different \( V_d \) compared to that of \( R_{new} \). Then, we consider \( S_{k+1} \). We check \( V_d \) of each of rules belonging to \( S_{k+1} \). Suppose there are \( m_{k+1} \) rules whose \( V_d \) match with that of \( R_{new} \), and \( n_k \) rules whose \( V_d \) do not match with that of \( R_{new} \). We define \( CF_{k+1} = \sum_{a=0}^{m_{k+1}} CF(S^a_{k+1}) - \sum_{b=0}^{n_k} CF(S^b_{k+1}) \), where \( S^a_{k+1} \) \((0 \leq a \leq m_{k+1})\) represents a rule belonging to \( S_i \) with same \( V_d \) as \( R_{new} \), and \( S^b_{k+1} \) \((0 \leq b \leq n_{k+1})\) represents a rule belonging to \( S_i \) with different \( V_d \) compared to that of \( R_{new} \). In this way we find \( CF_{k+2}, CF_{k+3}, \ldots, CF_{k+i} \) corresponding to \( S_{k+2}, S_{k+3}, \ldots, S_{k+i} \) respectively. So, the credibility factor for \( R_{new} \) is calculated using equation (iv).

\[
CF(R_{new}) = \frac{1}{(i−1)}(CF_w C^k_w + CF_{k+1} C^{k+1}_w + \ldots + CF_{k+i} C^{k+i}_w), \text{if } CF_j C^j_w > \alpha \text{ for } k \leq j \leq k+i
\]

\[\text{........... (iv)}\]

If \( CF(R_j) C^j_w \leq \alpha \) for some \( j \), then the value of \( CF(R_j) C^j_w = 0 \). If for all \( j \), we get \( CF(R_j) C^j_w \leq \alpha \), then \( CF(R_{new}) = 0 \). If the R.H.S of equation (iv) is greater than 1, then \( CF(R_{new}) = 1 \).

One thing to be mentioned here is that if a rule \( R_{new} \) is induced from two constituent rules \( R_{cont} \) and \( R_{cont2} \) and \( d \) is available in only one constituent rule \( R_{cont2} \) with the conditional weightage \( C_w \) for fact \( f_{cont2} \), then \( d \) would also be available in the decision part of rule \( R_{new} \) with same truth value as \( R_{cont2} \) if \( CF(R_{cont2})C^j_w > \alpha \). Otherwise, \( d \) will not be present as part of the decision for \( R_{new} \).

4.1.2 Data structure for the proposed lattice

We have proposed a new lattice based data structure which would be followed to develop the knowledge base for a medical expert system. The lattice based KB is a multi-level non-linear bottom-up data structure with each level containing a number of nodes to hold the lattice elements. Each level can be visualized as a special type of node called a ‘Lattice_Level’ node which contains four fields: a) a pointer to the immediate Successor ‘Lattice_Level’ node, b) the current level, c) total number of lattice elements in the current level, d) array of pointers to hold all the lattice elements in the current level. The top-most ‘Lattice_Level’ node has an empty immediate successor field. Each lattice element is represented by another type of node called ‘Lattice_Element’ which contains five fields: a) a boolean label for the node, b) the ‘condition’ part \( C \) of a rule, c) the ‘decision’ part \( D \) of the same rule, d) a pointer to point the first ‘Lattice_Element’ node of the predecessor list which is composed of related ‘Lattice_Element’ nodes residing at the immediate predecessor level, e) a pointer to point the first ‘Lattice_Element’ node of the successor list which is
composed of related Lattice_Element nodes residing at the immediate successor level. So, every Lattice_Element node corresponds to a rule of the form C → D.

Label of a Lattice_Element node is a means for easy identification and easy reference to a Lattice_Element node in KB. A label is a boolean string of n bits, if the lattice is of order n. The Lattice_Element node present at level 0 is a boolean string of n 0s. The Lattice_Element node present at level n is a boolean string of n 1s. As a lattice of order n has n Lattice_Element nodes corresponding to n atomic rules at level 1, the label of each of the nodes would contain only one 1 and (n–1) 0s. The label of the first Lattice_Element node from the left hand side of the lattice would be represented as 0n-1. The label of the next Lattice_Element node would be represented as 0n-210, which is obtained by left-shifting the previous label by 1 bit. If l_i is a Lattice_Element node corresponding to 1th atomic rule at level 1, the label of l_i is 0^(n+i)10, which can be obtained by left-shifting the label of l_1. In this way we proceed until we find the label of the right-most Lattice_Element node at level 1 as 10^n. The Lattice_Element nodes at level 2 would have exactly two 1s in their labels. More elaborately, the Lattice_Element node with label 0^n+1 and the Lattice_Element node with label 0^n-210 would form the Lattice_Element node at level 2 whose label is 0^n+211, i.e., the label 0^n+211 is the ‘Boolean OR’ combination of 0^n-1 and 0^n-210. So, in general, as a Lattice_Element node at level 2 is constructed using two Lattice_Element nodes at level 1, the label of each Lattice_Element node is obtained by ORing the labels of its constituent rules. Similarly, label of every Lattice_Element node at level i (1 ≤ i ≤ n) is obtained by ORing the labels of all its constituent Lattice_Element nodes at (i–1)th level of the lattice.

The lattice based data structure can be visualized as a bi-directed graph G = (V, E), where V consists of two types of nodes: Lattice_Level nodes L_i, and Lattice_Element nodes L_{a,b}. In a lattice based data structure of order n, there are (n+1) L_i nodes and 2^n L_{a,b} nodes. For each L_{a,b}-type node L_{a,b} at ith level (0 < i ≤ n and 1 ≤ j ≤ n), we create an adjacency list called ‘Predecessor_Adjacency_List’ to hold all the L_{a,b} nodes residing at the immediate predecessor level (i–1), that are attached to the node L_{a,b}. The fourth field of a Lattice_Element node holds the address of the first node of the corresponding Predecessor_Adjacency_List. Again, for each L_{a,b} node L_{a,b} at ith level (0 ≤ i < n), we create another adjacency list called ‘Successor_Adjacency_List’ to hold all the L_{a,b} nodes, at the immediate successor level (i+1), that are attached to the node L_{a,b}. The fifth field of a Lattice_Element node holds the address of the first node of the corresponding Successor_Adjacency_List. E consists of arcs between L_{a,b} nodes of two adjacent levels. No edges exist among the L_{a,b} nodes at the same level.

In a structured and concrete way, we define the lattice based KB of order n (n is the number of atomic rules) with (n+1) levels as a bottom-up non-linear data structure containing

i) a head node, which is different from a Lattice_Level node or a Lattice_Element node, to hold the total number of levels in the lattice and a pointer to hold the address of the Lattice_Level node at the 0th level.

ii) a Lattice_Level node at the 0th level to hold only a root Lattice_Element node whose second, third and fourth fields are empty. The Lattice_Element node has a Successor_Adjacency_List of n Lattice_Element nodes at 1st level.

iii) a Lattice_Level node at the nth level to hold only an end/terminal Lattice_Element node whose fifth field is empty. This Lattice_Level node has a Predecessor_Adjacency_List of n Lattice_Element nodes at (n–1)th level.

iv) a Lattice_Level node at the lth level (0 < l < (n+1)) to hold \( C_i \) Lattice_Element nodes with all the fields of each Lattice_Element node being non-empty. Each Lattice_Element node has a Successor_Adjacency_List of (n–l) Lattice_Element nodes and a Predecessor_Adjacency_List of l Lattice_Element nodes.

We now provide the structural definitions of different types of nodes used for our design using C language like constructs.

a. Structural Definition of head node

```c
struct head
{
    int count_level; // count_level indicates the total no. of levels present in the lattice
```
struct Lattice_Level_Node *ptr;  // ptr points to the Lattice_Level node at 0th level 
"
}

c. Structural Definition of a Lattice_Element node

struct Lattice_Element_Node
{
  bool label[];                  // label of a Lattice_Element node is represented by a boolean string
  char condition[];              // condition of a Lattice_Element_Node is basically a character string
  struct Decision *decision;     // decision is instantiation of a Decision structure
  struct Node *predecessor;      // predecessor is the pointer to the Predecessor_Adjacency_List
  struct Node *successor;        // successor is the pointer to the Successor_Adjacency_List
}

struct Decision
{
  struct Decs *D[] ;            // Decs structure is used for formation of the pair <d, V_d>
  int tot_decs;                // tot_decs keeps track of the total number of decisions for a Lattice_Element_Node
}

struct Decs
{
  char disease[];               // disease of a rule is a character string
  float Cond_Weight[N];        // Cond_Weight[i][1 ≤ i ≤ N] corresponds to the conditional weight of i-th fact for a decision, with N being the total number of constituent facts */
  float T1;                     // truth_value stands for V_d of d
  float T2;                     // CF represents the Credibility_Factor
  float T3;
  int truth_value;              // truth_value stands for V_d of d
  float CF;
}

struct Node
{
  bool label[];                 // label of a Lattice_Element node
  struct Node *ptr;            // ptr points to the next node in predecessor or successor adjacency list
}

In the following section, we give an algorithm for the knowledge base construction. We have used mixtures of natural languages and C language-like constructs for describing the algorithms.

4.1.3 Design of an algorithm for knowledge base construction

Procedure Build_KB (n)  // n is the no. of atomic rules
struct head *h = NIL;
struct Lattice_Level_Node *l, = NIL, *prev = NIL;
struct Lattice_Element_Node *ln = NIL;

/* Creation of Lattice_Level node at level 0 */
count_level [h] = n+1; //h keeps track the total number of levels in KB
level = 0;
l, = Create_Level_Node (prev, level) //The procedure Create_Level_Node generates a Lattice_Level node
For i = 0 to total_node_level [l,]−1 Do
    arr[l,] = Create_Level_Node (l,); /* The procedure Create_Level_Node generates a Lattice_Element node */
Endfor
ptr[h] = l,;

/* Creation of all the Lattice_Level nodes and corresponding Lattice_Element nodes */
level = level + 1;
prev = l,;
While (level ≠ count_level)
l, = Create_Level_Node (prev, level);
    For i = 0 to total_node_level [l,]−1 Do
        arr[l,] = Create_Level_Node (l,);
    Endfor
    prev ← l,;
    level ← level + 1;
Endwhile

/* Creation of adjacency lists of predecessor and successor nodes of each Lattice_Element node */
l, ← ptr[h];
While (l, ≠ NIL)
    For i = 0 to total_node_level [l,]−1 Do
        predecessor [arr[l,]] ← Create_Predecessor_List (arr[l,], level[l,]); /* The procedure Create_Predecessor_List creates Predecessor_Adjacency_List for a Lattice_Element_Node */
        successor [arr[l,]] ← Create_Successor_List (arr[l,], level[l,]); /* The procedure Create_Successor_List creates Successor_Adjacency_List for a Lattice_Element_Node */
    Endfor
    l, ← next_level[l,];
Endwhile

/* Returning the starting pointer of KB */
Return h;
End Build_KB

The procedure Create_Level_Node(prev, level) creates a new Lattice_Level_Node l, which resides as the next level of prev. The next_level field of l, is NIL and the total number of Lattice_Element nodes at level is “C_level. The procedure Create_Lattice_Node(l,) creates a new Lattice_Element node l, attached to l,. The label of l, is generated as per the process described earlier. Please refer to the Appendix A for detailed procedures.

Complexity analysis of Build_KB procedure

As per our proposed design, |V| = 2^s. So, n = log2 |V|. Suppose, the time taken by the procedure Create_Level_Node is T1 and that of the procedure Create_Lattice_Node is T2, where T1≠T2. The Create_Predecessor_List and Create_Successor_List procedures take O(log2 |V|) time each to execute. We assume that the (floor(n/2))th level holds N number of Lattice_Element nodes, where N = “C_node. In a lattice of order n, N is the maximum number of Lattice_Element nodes that a Lattice_Level node can hold. Therefore the time complexity of Build_KB algorithm is O(log2 |V|(|T1|+|T2|)). The algorithm is a polynomial-time algorithm. As the proposed design is basically a graph represented using adjacency list, the space complexity of the algorithm is O(|V|+|E|).
As our proposed design supports the construction of knowledge base based on the availability of domain knowledge at a particular time, the lattice based KB should have some scopes for insertion of additional knowledge, modification or deletion of some existing knowledge.

**Insertion of a new fact f into the existing lattice-based KB**

Insertion of a new fact can only be done at a new Lattice_Element_Node placed at the right-most end of the level 1 of the existing lattice KB. The label of the new Lattice_Element_Node is 10^n. With the insertion, the length of each label of each existing Lattice_Element_Node is increased by one. Except level 0, each Lattice_Level_Node l, at upper levels would now hold \( (n+1)^{C_{(\text{level}[l])}} \) Lattice_Element Nodes. So, at each level \( (n+1)^{C_{(\text{level}[l])}} - n^{C_{(\text{level}[l])}} \) new Lattice_Element_Nodes would be created at the right. A new level on the top of the existing KB would be created and this new level will have only one Lattice_Element node. Now, we have to modify the Predecessor_Adjacency_List and the Successor_Adjacency_List of every Lattice_Element_Node in the KB so that the updated KB follows a lattice structure of order \((n+1)\).

**Deletion of fact f from KB**

This operation deletes a fact f from all the Lattice_Element nodes which hold f in their condition parts. If a Lattice_Element_Node is deleted at level 1, then all its successors uptil the topmost level of KB would be deleted. The topmost level would also be deleted. Now the lattice is of order \((n - 1)\). Except level 0, each Lattice_Level_Node l, at upper levels would now hold \( n^{C_{(\text{level}[l])}} \) Lattice_Element Nodes. So, modifications should be made at the Predecessor_Adjacency_List and the Successor_Adjacency_List of every Lattice_Element_Node that is present in KB after the deletion of nodes. We have to now make some modifications in the labels of existing Lattice_Element nodes. For this, we check whether there is any ON bit (bit 1) present in the left substring with respect to the location of f (obviously the location of f contains a 0 bit) in the string of a label of a Lattice_Element node. If so, then there would be no change in the label. If there is any ON bit present in the right substring with respect to the location of f in the string, then we right-shift the ON bits by one place. Finally, we have to cut down the left-most 0 of every label.

**Modification of a Lattice_Element_Node in KB**

Modification of a Lattice_Element_Node means changing its condition or decision part. Change in condition part can be performed only at 1^{st} level. This change would be reflected into the condition parts of its successors at upper levels of the lattice. Change in condition part may or may not result in changes in decision part. If there is change in the decision part, then either a set of new decisions may be added to the existing decision part, or a set of existing decisions is replaced by a set of new decisions, or a set of existing decisions may be deleted. Changing a decision part includes modification of conditional weightages and credibility factors.

**4.2 Handling inconsistent knowledge using rough set**

The lattice-based KB may contain many inconsistent rules in which the condition parts imply inconsistent decisions. An inconsistent decision holds an inconclusive truth-value for a disease. Presence of inconsistent rules in a medical knowledge base is an obvious disadvantage as the medical expert system would not be able to infer diagnostic and therapeutic decisions accurately and reliably using these kinds of rules. With the aim of producing a dependable and reliable medical expert system, we use rough set theory to deal with the inconsistent rules.

**4.2.1 Approximation set generation for inconsistent knowledge in KB**

We consider our proposed lattice-based KB as an information system which may be expressed as \( S = (U, C, D) \), where \( U \) is a finite set of Lattice_Element nodes with distinct labels, \( C \) is a set of facts under consideration and \( D \) is the set of decisions. As we have said earlier, let us consider a Lattice_Element node \( l_n \) in KB which corresponds to an individual rule \( R_{l_n}: C_1 \rightarrow D_{l_n} \), which may not be optimal. There may be another rule in KB \( R_{l_m}: C_2 \rightarrow D_{l_m} \), where \( C_2 \subset C_1 \), and \( D_{l_m} = D_{l_n} \). As every Lattice_Element node has a label of a boolean string, we can say that every attribute/fact \( f \in C_1 \) is associated with a value of either 0 or 1. Every attribute \( d \in D \) is associated with a set of values from \( V_d \). As every label is distinct for every rule in KB, the indiscernibility relation \( I(C) \) would produce \( |U| \) singleton elementary sets. As there are \( |U| \) elementary sets, the reduct of the original set of attributes in KB is same as the set \( C \).
We have seen previously that there may be many *Lattice Element* nodes in the KB, which include one or more inconsistent decisions in their ‘decision’ parts. The easy method to identify inconsistent decision is to check whether $V_d$ is equal to 2. If so, then the decision is inconsistent. We define two concepts, where the first concept is called Concept$_1$ and holds the elementary sets associated with $V_d = 1$ for a particular decision $d$, and second concept is termed as Concept$_2$ which holds the elementary sets associated with $V_d = 0$ also for the decision $d$. As an inconclusive decision means that the disease may or may not be present, all the decisions of the rules with $V_d = 2$ must be included in both the concepts. Obviously, Concept$_1$ and Concept$_2$ contain both the consistent and inconsistent rules. We now calculate the lower and upper approximations of both the concept using the procedure *Approximation_Generation_KB*. The lower and upper approximation sets generated in this way also support the basic properties as stated by (Komorowski et al. 1999).

**Procedure Approximation_Generation_KB** $(h, d)$  
// $h$ is the starting address of the lattice-based KB

```plaintext
int l1 = 0, l2 = 0, l3 = 0;
ArrD = Search_Decision_KB(h, d); /* ArrD holds the Lattice_Element nodes which have $d$ as one of the decisions */
count = Size_of(ArrD); /* count is a temporary variable which holds the size of ArrD */

/* Generation of upper and lower approximations for Concept$_1$ */
For i = 0 to count − 1 Do
  If $V_d = 1$ OR $V_d = 2$ Then
    Upper_Approximation$_1[l1] = ArrD_i$; //Upper Approximation for Concept$_1$
    $l1 = l1 + 1$;
  Endif
  If $V_d = 1$ Then
    Lower_Approximation$_1[l2] = ArrD_i$; //Lower Approximation for Concept$_1$
    $l2 = l2 + 1$;
  Endif
  If $V_d = 2$ Then
    Boundary_Region$_1[l3] = ArrD_i$; //Boundary region for Concept$_1$
    $l3 = l3 + 1$;
  Endif
Endfor

/* Generation of upper and lower approximations for Concept$_2$ */
For i = 0 to count − 1 Do
  If $V_d = 0$ OR $V_d = 2$ Then
    Upper_Approximation$_2[l1] = ArrD_i$; //Upper Approximation for Concept$_2$
    $l1 = l1 + 1$;
  Endif
  If $V_d = 0$ Then
    Lower_Approximation$_2[l2] = ArrD_i$; //Lower Approximation for Concept$_2$
    $l2 = l2 + 1$;
  Endif
  If $V_d = 2$ Then
    Boundary_Region$_2[l3] = ArrD_i$; //Boundary region for Concept$_2$
    $l3 = l3 + 1$;
  Endif
Endfor

End Approximation_Generation_KB
```

The time complexity of the *Approximation_Generation_KB* algorithm is $O(\log_2|V|)$.
Now, we define how the sets of approximations can be modified when there are some changes in the KB. Our lattice-based design allows modification in approximation generation without generating the approximated sets again from scratch, thereby reducing the modification time. Modification of approximation generation can be done when a set of existing decisions are added to some other Lattice_Element nodes in KB, when some existing decisions are deleted from KB, and when truth values of some decisions are changed. Please refer to the Appendix B for detailed algorithm.

4.2.2 Optimal rule generation from KB

As per definitions of approximations of rough set, from the lower approximation of a concept, we find a set of certain rules. The rules generated from the upper approximation of a concept are called possible rules. The rules that are generated from the boundary region of a concept are called uncertain rules. Our proposed design also produces certain rules from the lower approximation sets generated for all the diseases present in KB. We also can get possible and uncertain rules from the upper approximation sets and boundary region sets respectively obtained from the lattice based information system.

As our rough set based lattice knowledge representation technique generates an exhaustive set of certain, possible, and uncertain rules given a set of facts, we may find many rules with common condition parts. Also, we may find a rule \( R_a \) with condition \( C_a \) and decision \( D \), and another rule \( R_b \) with condition \( C_b \) and decision \( D \) where \( C_a \subset C_b \). This would result in poor performance during the time of forward reasoning on the condition \( C_a \), as both the rules would be fired simultaneously. To avoid this kind of redundancy in the generated rule set, we propose an efficient rule minimization technique. The rules in the minimized rule set would hold only one decision in their decision parts.

**Minimization of rules**

We can minimize the rules using the concept of minimization of boolean functions in digital logic. We know that a boolean function can be expressed as a sum of minterms or product of maxterms. The process of simplifying the boolean expression of a Boolean function is called function minimization. The importance of minimization lies in reduction of cost and complexity of associated circuit. In this paper, we want to apply the function minimization technique to have minimum number of conditions against a decision. This would obviously reduce the inferencing time for a medical expert system. It is very much easy for the proposed design to apply function minimization procedure as we have labeled the Lattice_Element nodes using boolean strings.

Suppose, for a disease \( d \) to be present i.e. \( (V_d = 1) \), a set of rules \( S_R = \{ R_i \mid R_i \text{ is a rule and } 1 \leq i \leq n \} \) have been found from a lattice-based KB of order \( n \). Our assumption in this context is that if \( R_i \) is there, \( R_{i+1} \) may not be present in \( S_R \). If \( R_{i+1} \in S_R \) and \( R_{i+2} \in S_R \) where \( 1 \leq k_1, k_2 \leq n \), we can generate a minimized rule as \( \text{Condition}[R_{i+1}] + \text{Condition}[R_{i+2}] \rightarrow (d,1) \). This kind of consideration is similar to expressing the condition parts of elements of set \( S_R \) as sum of minterms. As each rule corresponds to a particular Lattice_Element node in the KB, the minterms are basically the labels of that particular nodes. Their sum is used to find out the minimum number of facts conditions. The minimized rule may end up with one or more conditions with or without any ‘OR’ operator among them. One thing to be mentioned here is that K-map procedure for boolean function minimization would not be applicable here as we have binary-coded the Lattice_Element nodes without using the actual binarization process.

As an example, we consider a lattice-based KB of order 4 with four different conditions \( c_1, c_2, c_3 \) and \( c_4 \) for four atomic rules available at level 1. Suppose a particular decision \((d,1)\) occurs for Lattice_Element nodes with labels 1000, 1001, 1101 and 1100. So, we define the sum of minterms as \( c_1 c_2 \varepsilon_1 + c_3 c_2 \varepsilon_1 + c_3 c_2 \varepsilon_1 + c_3 c_2 \varepsilon_1 \) which would be minimized as \( c_4 c_2 \). So, the minimized rule is: \( c_4 c_2 \rightarrow (d,1) \). The implication is that if a patient has symptom \( c_4 \) but he/she has no symptom of \( c_2 \), then we would infer that the disease \( d \) may be present for the patient. It is to be noted here that associated with \( c_4 \), a patient may have the other symptoms \( c_1 \) or/and \( c_3 \), but he/she does not have at all the symptom \( c_2 \). Let us take another example with the same lattice-based KB. We consider that another decision \((d_1,1)\) occurs for Lattice_Element nodes with labels 1100, 1101, 1001, 1111, 1011, 1110, and 1010. In this case, we define the sum of minterms as \( c_1 c_2 \varepsilon_1 + c_3 c_2 \varepsilon_1 + c_4 c_2 \varepsilon_1 + c_3 c_2 \varepsilon_1 + c_3 c_2 \varepsilon_1 + c_3 c_2 \varepsilon_1 \) which would be minimized as \( c_4 (c_1 + c_2 + c_1) \). So, the minimized rule is: \( c_4 (c_1 + c_2 + c_1) \rightarrow (d_1,1) \). The implication is that if a patient has symptom \( c_4 \) and associated with this, he/she has any other symptom from the set \{ \( c_1, c_2, c_1 \) \}, then we can infer that the disease \( d_1 \) may be present for the patient.
4.2.3 Defining certainty and coverage factors for generated optimal rules

In classical rough set theory, while calculating the certainty factor and coverage factor of a rule, we use all the reliable and unreliable data sources. As reliability is an important concern in medical expert systems, we would use only the reliable sources for calculating certainty and coverage factors of a rule generated from the lattice-based KB. We redefine the ‘Support’ of an optimal rule keeping in mind its credibility factor.

**Definition 1: Support** of an optimal rule generated from lattice based KB ($\$\$)

Suppose $R: C \rightarrow D$ is an optimal rule, which has been obtained by minimizing a set of rules namely $R_1; C_1 \rightarrow D, R_2; C_2 \rightarrow D, \ldots, R_k; C_k \rightarrow D (i,k>0)$; i.e. $C_1 \cap C_2 \cap \ldots \cap C_k = C$. We define the support ($\$\$) of each rule belonging to the set $\{R_1, R_2, \ldots, R_k\}$ as

$$s(R_j) = CF(R_j), \text{ where } 1 \leq j \leq k$$

Here $s(R_j)$ is called the ‘fractional support’ of $R_j$ and only takes into account the reliable knowledge sources, discarding the unreliable or unacceptable knowledge sources.

Now, we calculate the fractional support of the optimal rule $R$ as follows:

$$s(R) = \sum_{n=1}^{k} CF(R_n)$$

**Definition 2: Strength** of an optimal rule generated from lattice based KB ($\$\$)

We consider that, there exist $l (>0)$ rules such as $R_1; C_1 \rightarrow D, R_2; C_2 \rightarrow D, \ldots, R_l; C_l \rightarrow D$, where $\{R_1, R_2, \ldots, R_l\} \subseteq \{R_1, R_2, \ldots, R_k\}$. Please note that $V_d$ of all the rules belonging to the set $\{R_1, R_2, \ldots, R_l\}$ might not be same. From the set $\{R_1, R_2, \ldots, R_l\}$, we calculate $\Psi_D = \sum_{m=1}^{l} CF(R_m)$. Now, we define the strength ($\$\$) of each rule $R_j$ belonging to the set $\{R_1, R_2, \ldots, R_l\}$ as

$$\$\$ (R_j) = s(R_j)/\Psi_D, \text{ where } 1 \leq j \leq l$$

Here, $\$\$ (R_j)$ is called the ‘reliability strength’ of $R_j$. Now, we calculate the reliability strength of the optimal rule $R$ as follows:

$$\$\$ (R) = s(R)/\Psi_D$$

If $\{R_1, R_2, \ldots, R_l\}=\{R_1, R_2, \ldots, R_l\}$, then the strength of the rule $R$ would be 1. So, the strength of a rule will remain between 0 and 1.

**Definition 3: Certainty Factor** of an optimal rule generated from lattice based KB ($\$\$)

As per our preliminary assumption, the set of rules $\{R_1, R_2, \ldots, R_k\}$ hold $C$ in their condition parts with $V_d = x$. We first determine the set of rules $\{R_{h1}, R_{h2}, \ldots, R_{hl}\}$ from the set $\{R_1, R_2, \ldots, R_k\}$ where $V_d = x$. Obviously, $\{R_{h1}, R_{h2}, \ldots, R_{hl}\} \subseteq \{R_1, R_2, \ldots, R_k\}$. We determine $\Psi_C = \sum_{d=1}^{l} CF(R_{dj})$. We define the certainty factor ($\$\$) of each rule $R_t$ belonging to the set $\{R_1, R_2, \ldots, R_k\}$ as follows:

$$\$\$ (R_t) = s(R_t)/\Psi_C = \$\$ (R_t) / \Pi(C), \text{ where } \Pi(C) = \Psi_C/\Psi_D$$

So, $\$\$ (R) = s(R)/\Psi_C = \$\$ (R) / \Pi(C), \text{ where } \Pi(C) = \Psi_C/\Psi_D$

Now, let’s forget about the set $\{R_1, R_2, \ldots, R_k\}$. Suppose a set of rules $\{R_{j1}, R_{j2}, \ldots, R_{jl}\} (j, l>0)$ hold $C$ in their condition parts with $V_d = 2$. We then also search for the set of rules $\{R_{h1}, R_{h2}, \ldots, R_{hl}\}$ ($h, p>0$) from the set $\{R_1, R_2, \ldots, R_l\}$ where $V_d = 2$. In this case, we determine $\Psi_C = \sum_{d=1}^{l} CF(R_{dj})$. So, the certainty factor ($\$\$) of each rule $R_t$ belonging to the set $\{R_1, R_2, \ldots, R_k\}$ as follows:
\[ \mathfrak{Z}(R) = \frac{1}{2} \left[ \mathcal{S}(R) \cap \mathcal{C} \right] = \frac{1}{2} \left[ \mathcal{B}(R) / \Pi(C) \right], \text{ where } \Pi(C) = \Psi_C / \Psi_D \]

So, \[ \mathfrak{Z}(R) = \frac{1}{2} \left[ \mathcal{S}(R) \cap \mathcal{C} \right] = \frac{1}{2} \left[ \mathcal{B}(R) / \Pi(C) \right], \text{ where } \Pi(C) = \Psi_C / \Psi_D \]

Here the fraction ‘1/2’ comes because, for each rule \( R \) we get two conclusions in the consistent rule set, one with \( V_d = 1 \), and another with \( V_d = 0 \).

If \( \mathfrak{Z}(R) = 1 \), \( R \) is called a certain rule.

**Definition 4: Coverage Factor** of an optimal rule generated from lattice based KB (3)

With the same considerations as in Definition 3, we determine \( \mathfrak{Z}(\varphi_{(d = x)}) = \sum_{d=1}^{im} CF(R_d) \), where \( x \neq 2 \). We define the coverage factor (3) of each rule \( R_i \) belonging to the set \( \{R_1, R_2, \ldots, R_h\} \) as follows:

\[ \mathfrak{Z}(R_i) = \frac{\mathcal{S}(R_i)}{\varphi_{(d = x)}} = \mathcal{B}(R_i) / \Pi(V_d = x), \text{ where } \Pi(V_d = x) = \varphi_{(d = x)} / \Psi_D \]

So, \[ \mathfrak{Z}(R_i) = \frac{\mathcal{S}(R_i)}{\varphi_{(d = x)}} = \mathcal{B}(R_i) / \Pi(V_d = x), \text{ where } \Pi(V_d = x) = \varphi_{(d = x)} / \Psi_D \]

Now, let’s forget about the set \( \{R_1, R_2, \ldots, R_h\} \). Suppose a set of rules \( \{R_{j1}, R_{j2}, \ldots, R_{jm}\} (j, \ g > 0) \) hold \( C \) in their condition parts with \( V_d = 2 \). In this case, we determine \( \varphi_{(d = 2)} = \sum_{d=b+1}^{hp} CF(R_d) \). So, the coverage factor (3) of each rule \( R_j \) belonging to the set \( \{R_{j1}, R_{j2}, \ldots, R_{jm}\} \) as follows:

\[ \mathfrak{Z}(R_j) = \frac{1}{2} \left[ \mathcal{S}(R_j) / \varphi_{(d = 2)} \right] = \frac{1}{2} \left[ \mathcal{B}(R_j) / \Pi(V_d = 2) \right], \text{ where } \Pi(V_d = 2) = \varphi_{(d = 2)} / \Psi_D \]

So, \[ \mathfrak{Z}(R_j) = \frac{1}{2} \left[ \mathcal{S}(R_j) / \varphi_{(d = 2)} \right] = \frac{1}{2} \left[ \mathcal{B}(R_j) / \Pi(V_d = 2) \right], \text{ where } \Pi(V_d = 2) = \varphi_{(d = 2)} / \Psi_D \]

The coverage factor gives the conditional probability of reasons for a given decision.

**4.2.4 Probabilistic properties of the minimized rule set**

We know that any conventional decision table holds some basic probabilistic properties of rough set (Pawlak (2002)). We will now prove that the optimal decision rule set generated from lattice-based KB also holds some probabilistic properties.

**Property 1:** \[ \sum_{i=1}^{wK} \mathfrak{Z}(R_i) = 1 \]

**Proof:** Suppose, we have \( K \) (>0) rules namely \( R_{w1}, R_{w2}, \ldots, R_{wK} \) with conditions \( C_{w1}, C_{w2}, \ldots, C_{wK} \) respectively and decision \( D \). Please note that \( V_d \) of all the rules might not be same. Consider that \( C_{w1} \cap C_{w2} \cap \ldots \cap C_{wK} = C \). Obviously, \( \Psi_C = \sum_{d=w1}^{wK} CF(R_d) \). We now calculate the following:

\[ \sum_{i=1}^{wK} \mathfrak{Z}(R_i) = \frac{\mathcal{S}(R_{w1})}{\Psi_C} + \frac{\mathcal{S}(R_{w2})}{\Psi_C} + \ldots + \frac{\mathcal{S}(R_{wK})}{\Psi_C} \]

\[ = \frac{\mathcal{S}(R_{w1}) + \mathcal{S}(R_{w2}) + \ldots + \mathcal{S}(R_{wK})}{\Psi_C} \]

\[ = \sum_{d=w1}^{wK} CF(R_d)/\Psi_C = 1 \]

**Property 2:** \[ \sum_{m=x1}^{xJ} \mathfrak{Z}(R_m) = 1 \]

**Proof:** Suppose, we have \( J \) (>0) rules namely \( R_{x1}, R_{x2}, \ldots, R_{xJ} \) with decision \( D \) where \( V_d = x \) for all the rules (\( x \) may be either 0 or 1). Obviously, \( \varphi_{(d = x)} = \sum_{d=x1}^{xJ} CF(R_d) \). We now calculate the following:

\[ \sum_{m=x1}^{xJ} \mathfrak{Z}(R_m) = \frac{\mathcal{S}(R_{x1})}{\varphi_{(d = x)}} + \frac{\mathcal{S}(R_{x2})}{\varphi_{(d = x)}} + \ldots + \frac{\mathcal{S}(R_{xJ})}{\varphi_{(d = x)}} \]

\[ = \frac{\mathcal{S}(R_{x1}) + \mathcal{S}(R_{x2}) + \ldots + \mathcal{S}(R_{xJ})}{\varphi_{(d = x)}} \]

\[ = \sum_{d=x1}^{xJ} CF(R_d)/\varphi_{(d = x)} = 1 \]

**Property 3:** \( (\sum_{i=w1}^{wK} \mathfrak{Z}(R_i)) \cdot \Pi(C) = \sum_{i=w1}^{wK} \mathcal{B}(R_i) \)
Proof: When we consider the set \( \{R_{i1}, R_{i2}, \ldots, R_{ik}\} \) as per our preliminary assumption, \( \sum_{i=1}^{W} \zeta(R_i) = 1 \) according to property 1 and \( \Pi(C) = \Phi_c/\Phi_D \). Also, \( \vartheta(R_i) = \xi(R_i)/\Phi_D \). So, \( \sum_{i=1}^{W} \vartheta(R_i) = \Phi_c/\Phi_D \). So, it is proved that L.H.S = R.H.S.

Property 4: \( (\sum_{m=x1}^{x} \zeta(R_m)). \Pi(V_d = x) = \sum_{m=x1}^{x} \vartheta(R_m) \)

Proof: When we consider the set \( \{R_{i1}, R_{i2}, \ldots, R_{ij}\} \) as per our preliminary assumption, \( \sum_{m=x1}^{x} \zeta(R_m) = 1 \) according to property 2 and \( \Pi(V_d = x) = \vartheta(V_d = x)/\Phi_D \). Also, \( \vartheta(R_m) = \xi(R_m)/\Phi_D \). So, \( \sum_{m=x1}^{x} \zeta(R_m) \). So, it is proved that L.H.S = R.H.S.

Property 5: \( \forall R_i \in \{R_{i1}, R_{i2}, \ldots, R_{ik}\}, \zeta(R_i) = \frac{\vartheta(R_i)}{\sum_{i=1}^{W} \vartheta(R_i)} \)

Proof: We already know from definition 2 that \( \vartheta(R_i) = \xi(R_i)/\Phi_D \). \( \sum_{i=1}^{W} \vartheta(R_i) = \xi(R_{i1}) + \xi(R_{i2}) + \ldots + \xi(R_{ik}) \). So, \( \vartheta(R_i)/\Phi_D = \xi(R_i)/\Phi_D \). Also, \( \vartheta(R_i)/\Phi_D = \xi(R_i)/\Phi_D \). So, \( \vartheta(R_i)/\Phi_D = \xi(R_i)/\Phi_D \). So, it is proved that L.H.S = R.H.S.

Proof: The proof is similar to the proof of property 5. We know from definition 2 that \( \vartheta(R_i) = \xi(R_i)/\Phi_D \). Also, \( \vartheta(R_i) + \vartheta(R_j) + \vartheta(R_k) = \Phi(\Phi_{ldv} = v) \). So, \( \vartheta(R_i)/\Phi_D + \vartheta(R_j)/\Phi_D + \vartheta(R_k)/\Phi_D = \xi(R_i)/\Phi_D \). Also, \( \vartheta(R_i)/\Phi_D = \xi(R_i)/\Phi_D \). So, it is proved that L.H.S = R.H.S.

5. Sample case study from LBP domain

We illustrate our proposed knowledge representation scheme using a simple example from the LBP domain. The example taken here is not any clinically proved case study; rather a hypothetical situation, drawn on some primary knowledge about LBP, has been depicted for the sake of demonstration. As per our previous assumption, five categories of knowledge sources exist for LBP domain such as national guidelines (level 1), RCT (level 2), existing literature (level 3), patient case studies (level 4), and recommendations of expert physicians (level 5) category.

We consider three symptoms: LBP without leg pain, increased LBP at forward bending, LBP aggravated by prolonged sitting. We define three facts as follows: \( f_1 = \) (“LBP without leg pain”, “yes”), \( f_2 = \) (“increased LBP at forward bending”, “yes”), and \( f_3 = \) (“LBP aggravated by prolonged sitting”, “yes”). Suppose 8 knowledge sources of level 3 category exist which indicate that disease sacroiliac joint pain (SIJ) (Tilvawala et al. 2018; Visser et al. 2013) will be present certainly against \( f_1 \), 6 knowledge sources of level 5 category are there to indicate that SIJ will be certainly absent for \( f_1 \), and 7 knowledge sources of level 3 category as well as 5 sources of level 5 category exist to say that SIJ may or may not occur for \( f_1 \). We also consider that 10 knowledge sources of level 3 category exist to give evidence about the fact that CFJ would certainly occur against the symptom \( f_1 \), only 1 knowledge source is there which says that CFJ will not occur certainly for \( f_1 \), and 7 knowledge sources of level 5 category are available which state that CFJ may or may not occur for \( f_1 \). Discogenic pain may also occur certainly for \( f_1 \) (supported by 12 knowledge sources of level 3 category). It is also verified by 5 knowledge sources of level 4 category that discogenic pain will never occur for \( f_1 \). 13 knowledge sources of level 3 category and 7 knowledge sources of level 5 category show that discogenic pain may or may not occur for \( f_1 \). Whether \( f_1 \) would be liable for occurrence of disease MPS, is supported by a number of knowledge sources, among which 20 knowledge sources of level 3 category, 15 knowledge sources of level 4 category, and 10 sources of level 5 category claim that MPS would certainly occur for \( f_1 \). Only 2 knowledge sources of level 5 category state that MPS would certainly be absent from the set of possible diseases for \( f_1 \). 4 sources of level 4 category and 3 sources of level 5 category support the fact that MPS may or may not occur for \( f_1 \). To check whether prolapsed inter-vertebral disc (PIVD) (Fairbank (2008)) is there against \( f_1 \), we get no evidence for certain presence of the disease, but have 16 level 3 sources, 9 level 4 sources, and 11 level 5 sources which say that PIVD would certainly be absent for \( f_1 \). No evidence is also there to claim both the absence or presence of PIVD for \( f_1 \).

With a set of knowledge sources for \( f_1 \) in hand, we now calculate the following using equation (i). Here we use the notations \( T^m_{v}(f_1, \text{SIJ}) \), \( T^m_{v}(f_1, \text{CFJ}) \), \( T^m_{v}(f_1, \text{DP}) \), \( T^m_{v}(f_1, \text{MPS}) \), and \( T^m_{v}(f_1, \text{PIVD}) \) (1 ≤ m ≤ 3) to denote the acceptability of knowledge sources that find correspondence between the symptom \( f_1 \) and diseases SIJ, CFJ, discogenic pain, MPS, and PIVD respectively. We show all the calculations in Table 1a and Table 1b. All the calculated values are approximated with a precision of 0.01.
Table 1a: Measurement of acceptability of knowledge sources for SIJ, CFJ and discogenic pain against symptom $f_i$

| $T_v(f_i, SIJ)$ | $T_v(f_i, CFJ)$ | $T_v(f_i, DP)$ | $T_v(f_i, Discogenic Pain)$ |
|-----------------|-----------------|----------------|-----------------------------|
| $\frac{8 \times 3}{8 \times 3 + 6 \times 7 + 3 \times 5 + 6}$ | $\frac{10 \times 3 + 1 + 7}{10 \times 3 + 17 + 7}$ | $\frac{12 \times 3}{12 \times 3 + 5 \times 2 + 13 \times 3 \times 7 + 5 \times 2}$ | $\frac{12 \times 3}{12 \times 3 + 5 \times 2 + 13 \times 3 \times 7 + 5 \times 2}$ |
| $\approx 0.43$ | $\approx 0.79$ | $\approx 0.39$ | $\approx 0.50$ |

Table 1b: Measurement of acceptability of knowledge sources for MPS and PIVD against symptom $f_i$

| $T_v(f_i, MPS)$ | $T_v(f_i, PIVD)$ |
|-----------------|-----------------|
| $\frac{20 \times 3 + 15 \times 2 + 10}{20 \times 3 + 15 \times 2 + 10 + 2 + 4 \times 2 + 3}$ | $\frac{16 \times 3 + 9 \times 2 + 11}{16 \times 3 + 9 \times 2 + 11 + 0}$ |
| $\approx 0.88$ | $\approx 0.00$ |

Table 2a: Measurement of acceptability of knowledge sources for PIVD, SIJ against symptom $f_2$

| $T_v(f_2, PIVD)$ | $T_v(f_2, SIJ)$ |
|-----------------|----------------|
| $\frac{9 \times 3 + 1 \times 2 + 2 + 2 \times 4 + 2 + 5 \times 3 + 3 \times 2 + 1}{9 \times 3 + 1 \times 2 + 2 + 2 \times 4 + 2 + 5 \times 3 + 3 \times 2 + 1}$ | $\frac{17 \times 3 + 7 \times 2 + 3}{17 \times 3 + 7 \times 2 + 3 + 11 \times 2 + 1 + 9 \times 3 + 8 \times 2 + 6}$ |
| $\approx 0.49$ | $\approx 0.49$ |

| $T_v^2(f_2, PIVD)$ | $T_v^2(f_2, SIJ)$ |
|-----------------|----------------|
| $\frac{9 \times 3 + 1 \times 2 + 2 + 2 \times 4 + 2 + 5 \times 3 + 3 \times 2 + 1}{9 \times 3 + 1 \times 2 + 2 + 2 \times 4 + 2 + 5 \times 3 + 3 \times 2 + 1}$ | $\frac{17 \times 3 + 7 \times 2 + 3}{17 \times 3 + 7 \times 2 + 3 + 11 \times 2 + 1 + 9 \times 3 + 8 \times 2 + 6}$ |
| $\approx 0.16$ | $\approx 0.16$ |

Table 2b: Measurement of acceptability of knowledge sources for CFJ against symptom $f_2$

| $T_v(f_2, CFJ)$ |
|----------------|
| $\frac{1 \times 3 + 1 \times 2}{1 \times 3 + 1 \times 2 + 2 + 2 \times 4 + 1 + 7 \times 3 + 10 \times 2 + 2}$ |
| $\approx 0.09$ |

| $T_v^2(f_2, CFJ)$ |
|----------------|
| $\frac{1 \times 3 + 1 \times 2 + 2 + 2 \times 4 + 1 + 7 \times 3 + 10 \times 2 + 2}{1 \times 3 + 1 \times 2 + 2 + 2 \times 4 + 1 + 7 \times 3 + 10 \times 2 + 2}$ |
| $\approx 0.09$ |

| $T_v^3(f_2, CFJ)$ |
|----------------|
| $\frac{1 \times 3 + 1 \times 2 + 2 + 2 \times 4 + 1 + 7 \times 3 + 10 \times 2 + 2}{1 \times 3 + 1 \times 2 + 2 + 2 \times 4 + 1 + 7 \times 3 + 10 \times 2 + 2}$ |
| $\approx 0.81$ |
Finally, we consider symptom $f_3$. Suppose 6 knowledge sources of level 3 category and 3 sources of level 4 category exist which indicate that disease CFJ will be present certainly against $f_3$, 5 knowledge sources of level 3 category, 2 sources of level 4 category, and 2 sources of level 5 category are there to indicate that CFJ will be certainly absent for $f_3$, and 7 knowledge sources of level 3 category as well as 9 sources of level 4 category and 3 sources of level 5 category exist to say that CFJ may or may not occur for $f_3$. We also consider that 15 knowledge sources of level 3 category, 10 sources of level 4 category, and 11 sources of level 5 category exist to give evidence about the fact that PIVD would certainly occur against the symptom $f_3$, 8 knowledge sources of level 3 category, 8 knowledge sources of level 4 category, and 7 sources of level 5 category are there which say that PIVD will not occur $\pi r^2$ certainly for $f_3$, and 18 knowledge sources of level 3 category, 12 sources of level 4, and 7 sources of level 5 category are available which state that PIVD may or may not occur for $f_3$. SIJ may also occur certainly for $f_3$ (supported by 7 knowledge sources of level 3 category, 8 sources of level 4 category, and 3 sources of level 5 category). It is also verified by 3 knowledge sources of level 5 category that SIJ will never occur for $f_3$. 17 knowledge sources of level 3 category, 8 sources of level 4 category, and 5 knowledge sources of level 5 category show that SIJ may or may not occur for $f_3$. Whether $f_3$ would be liable for occurrence of discogenic pain, is supported by a number of knowledge sources, among which 15 knowledge sources of level 3 category, 5 knowledge sources of level 4 category, and 6 sources of level 5 category claim that discogenic pain would certainly occur for $f_3$. 6 knowledge sources of level 3 category, another 6 sources of level 4 category, and 3 sources of level 5 category state that discogenic pain would certainly be absent for the set of possible diseases for $f_3$. 7 sources of level 4 category and 4 sources of level 5 category support the fact that discogenic pain may or may not occur for $f_3$.

With a set of knowledge sources for $f_3$ in hand, we now calculate the following using equation (i). Here we use the notations $T_v^m(f_3, CFJ)$, $T_v^m(f_3, PIVD)$, $T_v^m(f_3, SIJ)$, and $T_v^m(f_3, DP)$ ($1 \leq m \leq 3$) to denote the acceptability of knowledge sources that find correspondence between the symptom $f_3$ and diseases CFJ, PIVD, SIJ, and discogenic pain respectively. We show all the calculations in Table 3a and Table 3b. All the calculated values are approximated with a precision of 0.01.

### Table 3a: Measurement of acceptability of knowledge sources for CFJ, PIVD against symptom $f_3$

|          | $<f_3, CFJ>$ | $<f_3, PIVD>$ |
|----------|-------------|---------------|
| $T_v^1(f_3, CFJ)$ = | $\frac{6 \times 3 + 3 \times 2}{6 \times 3 + 3 \times 2 + 5 \times 3 + 2 \times 2 + 7 \times 3 + 9 \times 2 + 3}$ | $\approx 0.28$ | $T_v^1(f_3, PIVD)$ = | $\frac{15 \times 3 + 10 \times 2 + 11}{15 \times 3 + 10 \times 2 + 11 + 8 \times 3 + 8 \times 2 + 7 + 18 \times 3 + 12 \times 2 + 7}$ | $\approx 0.37$ |
| $T_v^2(f_3, CFJ)$ = | $\frac{6 \times 3 + 3 \times 2 + 5 \times 3 + 2 \times 2 + 7 \times 3 + 9 \times 2 + 3}{6 \times 3 + 3 \times 2 + 5 \times 3 + 2 \times 2 + 7 \times 3 + 9 \times 2 + 3}$ | $\approx 0.24$ | $T_v^2(f_3, PIVD)$ = | $\frac{15 \times 3 + 10 \times 2 + 11 + 8 \times 3 + 8 \times 2 + 7 + 18 \times 3 + 12 \times 2 + 7}{15 \times 3 + 10 \times 2 + 11 + 8 \times 3 + 8 \times 2 + 7 + 18 \times 3 + 12 \times 2 + 7}$ | $\approx 0.23$ |
| $T_v^3(f_3, CFJ)$ = | $\frac{6 \times 3 + 3 \times 2 + 5 \times 3 + 2 \times 2 + 7 \times 3 + 9 \times 2 + 3}{6 \times 3 + 3 \times 2 + 5 \times 3 + 2 \times 2 + 7 \times 3 + 9 \times 2 + 3}$ | $\approx 0.48$ |

### Table 3b: Measurement of acceptability of knowledge sources for SIJ, Discogenic Pain against symptom $f_3$

|          | $<f_3, SIJ>$ | $<f_3, Discogenic Pain>$ |
|----------|-------------|--------------------------|
| $T_v^1(f_3, SIJ)$ = | $\frac{7 \times 3 + 8 \times 3}{7 \times 3 + 8 \times 3 + 3 \times 17 + 3 \times 9 \times 2 + 5}$ | $\approx 0.35$ | $T_v^1(f_3, DP)$ = | $\frac{15 \times 3 + 5 \times 2 + 6 \times 6 + 3 \times 6 + 2 + 3 + 7 \times 2 + 4}{6 \times 3 + 6 \times 2 + 3 + 7 \times 7 \times 2 + 4}$ | $\approx 0.54$ |
| $T_v^2(f_3, SIJ)$ = | $\frac{7 \times 3 + 8 \times 3 + 3 \times 17 + 3 \times 9 \times 2 + 5}{7 \times 3 + 8 \times 3 + 3 \times 17 + 3 \times 9 \times 2 + 5}$ | $\approx 0.03$ | $T_v^2(f_3, DP)$ = | $\frac{15 \times 3 + 5 \times 2 + 6 \times 6 + 3 \times 6 + 2 + 3 + 7 \times 2 + 4}{15 \times 3 + 5 \times 2 + 6 \times 6 + 3 \times 6 + 2 + 3 + 7 \times 2 + 4}$ | $\approx 0.29$ |
| $T_v^3(f_3, SIJ)$ = | $\frac{7 \times 3 + 8 \times 3 + 3 \times 17 + 3 \times 9 \times 2 + 5}{7 \times 3 + 8 \times 3 + 3 \times 17 + 3 \times 9 \times 2 + 5}$ | $\approx 0.63$ | $T_v^3(f_3, DP)$ = | $\frac{15 \times 3 + 5 \times 2 + 6 \times 6 + 3 \times 6 + 2 + 3 + 7 \times 2 + 4}{15 \times 3 + 5 \times 2 + 6 \times 6 + 3 \times 6 + 2 + 3 + 7 \times 2 + 4}$ | $\approx 0.16$ |

We now calculate $V_0$ individually for each of the diseases of a symptom. For this, we have to construct a $(3 \times 5)$ matrix for each symptom-disease pair. We denote $K_1$, $K_2$ to represent all the knowledge sources that claim certain presence of a disease corresponding to a symptom, and certain absence of the disease for a symptom respectively. We represent $K_3$ to indicate that the disease may or may not be present for the symptom. The rows hold $K_1$, $K_2$, and $K_3$, while the columns represent level-wise distribution of knowledge sources. Table 4 represents the $M$ matrix (refer to section 4.1) for the pair $<f_i, SIJ>$, denoted by $M(f_i, SIJ)$.

### Table 4: Design of matrix $M(f_i, SIJ)$ for calculation of $V_0$ for the pair $<f_i, SIJ>$

|                | level 1 Category | level 2 Category | level 3 Category | level 4 Category | level 5 Category |
|----------------|------------------|------------------|------------------|------------------|------------------|
| $K_1$          | 0                | 0                | 1                | 0                | 0                |
As per our strategy for determining \( V_d \) for the pair \(<f_i, SIJ>\) from \( M(f_i, SIJ) \), we calculate \( V_d = 2 \). Here we say that the pair \(<f_i, SIJ>\) takes part in a rule \( R_{1a} \). So, \( R_{1a} \) is an atomic rule. From the matrix \( M(f_i, SIJ) \) and the corresponding \( T_v^m(f_j, SIJ) \), we calculate \( CF = 0.46 \). We construct similar matrices for the other symptom-disease pairs. We construct a rule \( R_{1b} \) to hold \(<f_i, CFJ>\) with \( V_d = 1 \), a rule \( R_{1c} \) to hold \(<f_i, Discogenic Pain>\) with \( V_d = 2 \), a rule \( R_{1d} \) to hold \(<f_i, MPS>\) with \( V_d = 1 \), a rule \( R_{1e} \) to hold \(<f_i, PIVD>\) with \( V_d = 0 \). We calculate \( CF(R_{1b}) = 0.79 \), \( CF(R_{1c}) = 0.50 \), \( CF(R_{1d}) = 0.88 \), and \( CF(R_{1e}) = 1.00 \). All the rules are atomic rules.

We follow the same procedure for determining \( V_d \) and \( CF \) for atomic rules \( R_{2a} \) (\(<f_2, PIVD>\), \( R_{2b} \) (\(<f_2, SIJ>\), and \( R_{2c} \) (\(<f_2, CFJ>\)). For \( R_{2a}, V_d = 1 \) and \( CF(R_{2a}) = 0.49 \); for \( R_{2b}, V_d = 1 \) and \( CF(R_{2b}) = 0.49 \); for \( R_{2c}, V_d = 2 \) and \( CF(R_{2c}) = 0.81 \). Also, the same procedure is followed for determining \( V_d \) and \( CF \) for atomic rules \( R_{3a} \) (\(<f_3, CFJ>\), \( R_{3b} \) (\(<f_3, PIVD>\), \( R_{3c} \) (\(<f_3, SIJ>\), and \( R_{3d} \) (\(<f_3, Discogenic Pain>\)). For \( R_{3a}, V_d = 2 \) and \( CF(R_{3a}) = 0.48 \); for \( R_{3b}, V_d = 2 \) and \( CF(R_{3b}) = 0.41 \); for \( R_{3c}, V_d = 2 \) and \( CF(R_{3c}) = 0.63 \); for \( R_{3d}, V_d = 1 \) and \( CF(R_{3d}) = 0.54 \).

Based on the three facts \( f_1, f_2, \) and \( f_3 \), we construct a lattice structure that is described in section 4.1. Figure 2 gives the Hasse diagram based on which the data structure (section 4.1.2) is designed.

**Figure 2.** Lattice construction based on facts \( f_1, f_2, \) and \( f_3 \)

We construct composite rules for level 2 of the lattice structure depicted in Figure 2. Suppose \( R_{12a} \) is a composite rule to hold condition \((f_1 \ AND \ f_2)\) and disease SIJ, with conditional weightages \( 1/2 \) and \( 1/2 \) for \( f_1 \) and \( f_2 \) respectively; \( R_{12b} \) is a composite rule to hold condition \((f_1 \ AND \ f_2)\) and disease CFJ, with conditional weightages \( 2/3 \) and \( 1/3 \) for \( f_1 \) and \( f_2 \) respectively; \( R_{12c} \) is a composite rule to hold condition \((f_1 \ AND \ f_2)\) and disease discogenic pain, with conditional weightages \( 2/3 \) and \( 1/3 \) for \( f_1 \) and \( f_2 \) respectively; \( R_{12d} \) is a composite rule to hold condition \((f_1 \ AND \ f_2)\) and disease MPS, with conditional weightages \( 2/3 \) and \( 1/3 \) for \( f_1 \) and \( f_2 \) respectively; \( R_{12e} \) is a composite rule to hold condition \((f_1 \ AND \ f_2)\) and disease PIVD with conditional weightages \( 1/2 \) and \( 1/2 \) for \( f_1 \) and \( f_2 \) respectively. Observing carefully the atomic rules at level 1 of lattice which hold either \( f_1 \) or \( f_2 \), we get \( V_d = 1 \) for \( R_{12a} \) and \( CF(R_{12a}) = (0.49 \times 0.50 - 0.46 \times 0.50) = 0.02 \); \( V_d = 2 \) for \( R_{12b} \) and \( CF(R_{12b}) = (0.79 \times 2/3 - 0.81 \times 1/3) \approx 0.26 \); \( V_d = 2 \) for \( R_{12c} \) and \( CF(R_{12c}) = (0.50 \times 2/3) \approx 0.33 \); \( V_d = 1 \) for \( R_{12d} \) and \( CF(R_{12d}) = (0.88 \times 2/3) \approx 0.59 \); \( V_d = 0 \) for \( R_{12e} \) and \( CF(R_{12e}) = (1.00 \times 1/2 - 0.49 \times 1/2) \approx 0.26 \).

Suppose \( R_{13a} \) is another composite rule to hold condition \((f_1 \ AND \ f_3)\) and disease SIJ, with conditional weightages \( 1/2 \) and \( 1/2 \) for \( f_1 \) and \( f_3 \) respectively; \( R_{13b} \) is a composite rule to hold condition \((f_1 \ AND \ f_3)\) and disease CFJ, with conditional weightages \( 1/2 \) and \( 1/2 \) for \( f_1 \) and \( f_3 \) respectively; \( R_{13c} \) is a composite rule to hold condition \((f_1 \ AND \ f_3)\) and disease discogenic pain, with conditional weightages \( 2/3 \) and \( 1/3 \) for \( f_1 \) and \( f_3 \) respectively; \( R_{13d} \) is a composite rule to hold condition \((f_1 \ AND \ f_3)\) and disease MPS, with conditional weightages \( 2/3 \) and \( 1/3 \) for \( f_1 \) and \( f_3 \) respectively; \( R_{13e} \) is a composite rule to hold condition \((f_1 \ AND \ f_3)\) and disease PIVD with conditional weightages \( 2/3 \) and \( 1/3 \) for \( f_1 \) and \( f_3 \) respectively. Observing carefully the atomic rules at level 1 of lattice which hold either \( f_1 \) or \( f_3 \), we get \( V_d = 2 \) for \( R_{13a} \) and \( CF(R_{13a}) = (0.46 \times 0.50 + 0.63 \times 0.50) \approx 0.55 \); \( V_d = 1 \) for \( R_{13b} \) and \( CF(R_{13b}) = (0.79 \times 0.50 - 0.48 \times 0.50) \approx 0.16 \); \( V_d = 1 \) for \( R_{13c} \) and \( CF(R_{13c}) = (0.50 \times 2/3 - 0.54 \times 1/3) \approx 0.15 \); \( V_d = 2 \) for \( R_{13d} \) and \( CF(R_{13d}) = (1 \times 2/3 - 0.41 \times 1/3) \approx 0.53 \); \( V_d = 1 \) for \( R_{13e} \) and \( CF(R_{13e}) = (0.88 \times 2/3) = 0.59 \).
Consider another composite rule $R_{23a}$ to hold condition ($f_2 \text{ AND } f_3$) and disease SIJ, with conditional weightages 1/2 and 1/2 for $f_2$ and $f_3$ respectively; $R_{23b}$ is a composite rule to hold condition ($f_2 \text{ AND } f_1$) and disease CFJ, with conditional weightages 1/2 and 1/2 for $f_2$ and $f_1$ respectively; $R_{23c}$ is a composite rule to hold condition ($f_2 \text{ AND } f_3$) and disease discogenic pain, with conditional weightages 1/2 and 1/2 for $f_2$ and $f_3$ respectively; $R_{23d}$ is a composite rule to hold condition ($f_1 \text{ AND } f_2 \text{ AND } f_3$) and disease MPS, with conditional weightages 1/3, 1/3, and 1/3 for $f_1$, $f_2$, and $f_3$ respectively; $R_{123a}$ is a composite rule to hold condition ($f_1 \text{ AND } f_2 \text{ AND } f_3$) and disease CFJ, with conditional weightages 1/3, 1/3, and 1/3 for $f_1$, $f_2$, and $f_3$ respectively; $R_{123c}$ is a composite rule to hold condition ($f_1 \text{ AND } f_2 \text{ AND } f_3$) and disease discogenic pain, with conditional weightages 2/4, 1/4, and 1/4 for $f_1$, $f_2$, and $f_3$ respectively; $R_{123b}$ is a composite rule to hold condition ($f_1 \text{ AND } f_2 \text{ AND } f_3$) and disease MPS, with conditional weightages 1/3, 1/3, and 1/3 for $f_1$, $f_2$, and $f_3$ respectively. Observing carefully the atomic rules at level 1 of lattice which hold either $f_2$ or $f_3$, we get $V_d = 2$ for $R_{23a}$ and $CF(R_{23a}) = (0.63 \times 0.50 - 0.49 \times 0.50) \approx 0.07$; $V_d = 2$ for $R_{23b}$ and $CF(R_{23b}) = (0.81 \times 0.50 + 0.48 \times 0.50) \approx 0.65$; $V_d = 1$ for $R_{23c}$ and $CF(R_{23c}) = (0.49 \times 0.50 - 0.41 \times 0.50) \approx 0.04$; $V_d = 1$ for $R_{123a}$ and $CF(R_{123a}) = (0.54 \times 0.50) \approx 0.27$.

Now we construct a composite rule $R_{123a}$ to hold condition ($f_1 \text{ AND } f_2 \text{ AND } f_3$) and disease SIJ, with conditional weightages 1/3, 1/3, and 1/3 for $f_1$, $f_2$, and $f_3$ respectively; $R_{123b}$ is a composite rule to hold condition ($f_1 \text{ AND } f_2$) and disease CFJ, with conditional weightages 1/3, 1/3, and 1/3 for $f_1$, $f_2$, and $f_3$ respectively; $R_{123c}$ is a composite rule to hold condition ($f_1 \text{ AND } f_2 \text{ AND } f_3$) and disease discogenic pain, with conditional weightages 2/5, 1/5, and 2/5 for $f_1$, $f_2$, and $f_3$ respectively; $R_{123d}$ is a composite rule to hold condition ($f_1 \text{ AND } f_2 \text{ AND } f_3$) and disease MPS, with conditional weightages 1/3, 1/3, and 1/3 for $f_1$, $f_2$, and $f_3$ respectively. Observing carefully all the composite rules at level 2 of lattice, we get $V_d = 2$ for $R_{123a}$ and $CF(R_{123a}) = 1/2 \times [(0.55 - 0.02) \times 1/3 + (0.07 - 0.02) \times 1/3 + (0.55 + 0.07) \times 1/3] \approx 0.21$; $V_d = 2$ for $R_{123b}$ and $CF(R_{123b}) = 1/2 \times [(0.26 - 0.16) \times 1/3 + (0.26 + 0.65) \times 1/3 + (0.65 - 0.16) \times 1/3] \approx 0.25$; $V_d = 2$ for $R_{123c}$ and $CF(R_{123c}) = 1/2 \times [(0.53 - 0.26) \times 2/4 + (0.53 - 0.04) \times 1/4 + (0.26 - 0.04) \times 1/4] \approx 0.16$; $V_d = 2$ for $R_{123d}$ and $CF(R_{123d}) = 1/2 \times [(0.33 - 0.15) \times 2/5 + (0.15 + 0.27) \times 2/5 + (0.33 - 0.27) \times 1/5] \approx 0.13$; $V_d = 1$ for $R_{123e}$ and $CF(R_{123e}) = 1/2 \times [(0.59 + 0.59) \times 1/3 + 0.59 \times 1/3 + 0.59 \times 1/3] \approx 0.40$.

So, now we have a set of 31 un-optimized rules, some of which also contain inconsistent knowledge. We would use rough set theory to generate optimal rules after handling the inconsistencies in knowledge. In this example, we have five LBP diseases namely SIJ, CFJ, PIVD, discogenic pain, and MPS. For each of diseases we will find two concepts as described in section 4.2.1, which are given below:

Concept$_1$(SIJ) = \{R$_{1a}$, R$_{2b}$, R$_{3c}$, R$_{12a}$, R$_{13a}$, R$_{23a}$, R$_{123a}$\}, Concept$_2$(SIJ) = \{R$_{1a}$, R$_{3c}$, R$_{13a}$, R$_{23a}$, R$_{123a}$\}

Concept$_1$(CFJ) = \{R$_{1b}$, R$_{2c}$, R$_{3a}$, R$_{12b}$, R$_{13b}$, R$_{23b}$, R$_{123b}$\}, Concept$_2$(CFJ) = \{R$_{2c}$, R$_{3a}$, R$_{12b}$, R$_{23b}$, R$_{123b}$\}

Concept$_1$(PIVD) = \{R$_{2a}$, R$_{3b}$, R$_{13a}$, R$_{23c}$, R$_{123c}$\}, Concept$_2$(PIVD) = \{R$_{1a}$, R$_{3b}$, R$_{12b}$, R$_{3c}$, R$_{123c}$\}

Concept$_1$(Discogenic pain) = \{R$_{1c}$, R$_{3d}$, R$_{12c}$, R$_{13d}$, R$_{23d}$, R$_{123d}$\}, Concept$_2$(Discogenic pain) = \{R$_{1c}$, R$_{12c}$, R$_{123d}$\}

Concept$_1$(MPS) = \{R$_{1d}$, R$_{12d}$, R$_{13d}$, R$_{23d}$\}, Concept$_2$(MPS) = \{\}

As, Concept$_1$(SIJ) ∩ Concept$_2$(SIJ) ≠ \{\}, we have to determine the lower and upper approximations of both the concepts. Similarly, for Concept$_1$(CFJ) and Concept$_2$(CFJ), Concept$_1$(PIVD) and Concept$_2$(PIVD), Concept$_1$(Discogenic pain) and Concept$_2$(Discogenic pain), we have to determine the lower and upper approximations as per Approximation_Generation_KB procedure.

Lower approximation of Concept$_1$(SIJ) = \{R$_{2b}$, R$_{12a}$\}
Upper approximation of Concept$_1$(SIJ) = \{R$_{1a}$, R$_{2b}$, R$_{3c}$, R$_{12a}$, R$_{13a}$, R$_{23a}$, R$_{123a}$\}

Boundary region of Concept$_1$(SIJ) = \{R$_{1a}$, R$_{3c}$, R$_{13a}$, R$_{23a}$, R$_{123a}$\}
Lower approximation of Concept$_2$(SIJ) = \{\}
Upper approximation of Concept$_2$(SIJ) = \{R$_{1a}$, R$_{3c}$, R$_{13a}$, R$_{23a}$, R$_{123a}$\}

Lower approximation of Concept$_1$(CFJ) = \{R$_{1b}$, R$_{13b}$\}
Upper approximation of Concept$_1$(CFJ) = \{R$_{1b}$, R$_{3c}$, R$_{3a}$, R$_{12b}$, R$_{13b}$, R$_{23b}$, R$_{123b}$\}

Boundary region of Concept$_1$(CFJ) = \{R$_{2c}$, R$_{3a}$, R$_{12b}$, R$_{23b}$, R$_{123b}$\}
Lower approximation of Concept$_2$(CFJ) = \{\}
Upper approximation of Concept$_2$(CFJ) = \{R$_{2c}$, R$_{3a}$, R$_{12b}$, R$_{23b}$, R$_{123b}$\}

Lower approximation of Concept$_1$(PIVD) = \{R$_{2a}$, R$_{23c}$\}
Upper approximation of Concept₁(PIVD) = \{R_{2a}, R_{3b}, R_{13d}, R_{23c}, R_{123e}\}
Boundary region of Concept₁(PIVD) = \{R_{3b}, R_{13d}, R_{123e}\}
Lower approximation of Concept₁(PIVD) = \{R_{1c}, R_{12c}\}
Upper approximation of Concept₂(PIVD) = \{R_{1c}, R_{3b}, R_{12c}, R_{13d}, R_{123e}\}
Boundary region of Concept₂(PIVD) = \{R_{3b}, R_{13d}, R_{123e}\}

Lower approximation of Concept₁(Discogenic pain) = \{R_{3b}, R_{13c}, R_{23d}\}
Upper approximation of Concept₁(Discogenic pain) = \{R_{1c}, R_{3b, 12c}, R_{13c}, R_{23d}, R_{123d}\}
Boundary region of Concept₁(Discogenic pain) = \{R_{1c}, R_{12c}, R_{123d}\}
Lower approximation of Concept₂(Discogenic pain) = \emptyset
Upper approximation of Concept₂(Discogenic pain) = \{R_{1c}, R_{12c}, R_{123d}\}
Boundary region of Concept₂(Discogenic pain) = \{R_{1c}, R_{12c}, R_{123d}\}

Now, we generate certain and optimal decision rules using the lower approximations of concepts and applying the rule-minimization technique.

Rule 1. (“LBP aggravated by prolonged sitting”, “no”) AND (“increased LBP at forward bending”, “yes”) → (SIJ, 1) with reliability strength = 0.12.

Rule 2. (“LBP without leg pain”, “yes”) AND (“increased LBP at forward bending”, “no”) → (CFJ, 1) with reliability strength = 0.16.

Rule 3. (“LBP without leg pain”, “no”) AND (“increased LBP at forward bending”, “yes”) → (PIVD, 1) with reliability strength = 0.20.

Rule 4. (“LBP without leg pain”, “yes”) AND (“LBP aggravated by prolonged sitting”, “no”) → (PIVD, 0) with reliability strength = 0.29.

Rule 5. (“Increased LBP at forward bending”, “no”) AND (“LBP aggravated by prolonged sitting”, “yes”) → (Discogenic pain, 1) with reliability strength = 0.24.

Rule 6. (“LBP without leg pain”, “no”) AND (“LBP aggravated by prolonged sitting”, “yes”) → (Discogenic pain, 1) with reliability strength = 0.28.

Rule 5 and Rule 6 can be further minimized as: (“LBP aggravated by prolonged sitting”, “yes”) AND (“LBP without leg pain”, “no”) OR (“Increased LBP at forward bending”, “no”) → (Discogenic pain, 1)

Rule 7. (“LBP without leg pain”, “yes”) → (MPS, 1) with reliability strength = 1.00.

Here, we do not show the uncertain optimal rules as a clinical diagnosis should not allow, in most cases, the uncertain rules. The certain rules for a particular disease should be arranged in descending order according to their reliability strengths, i.e., the rule whose strength is high would be applicable first in case of backward reasoning. If no diagnostic decision is taken based on the certain rules, only then the uncertain rules are taken into account after consulting with domain experts.

6. Discussion of properties of proposed knowledge representation

Our proposed lattice-based knowledge representation scheme follows a number of properties as follows.

Completeness
Using our proposed knowledge representation, we can design a complete knowledge base KB as, based on n atomic rules, we can generate an exhaustive set of \((2^n - n - 1)\) composite rules. No other rules are possible from the set of atomic rules present in a knowledge module.

Consistency
The proposed scheme is consistent as we have generated the lower and upper approximations of concepts for each inconsistent decision. Also, in case of modification of the rules available in KB, we have kept provisions for generations of approximated sets if any kind of inconsistency is seen in the modified KB.

**Integrity**
The use of lattice structure for the design of KB allows to support integrity among the related rules in a knowledge module. As we have used the concept of ‘glb’ and ‘lub’ of a lattice, the condition part and decision part of a rule are designed based on the conditions and decisions of corresponding predecessor rules. Modification in KB also is performed based on the predecessor and successor nodes of a particular Lattice_Element node. So, the KB works in an integrated manner.

**Zero Redundancy**
The proposed design does not support redundant information, as we have incorporated lattice theory in our structure. As lattice theory does not allow duplicate set elements, there are no redundant rules in the KB. Also, applying the rule minimization technique on the decision rules generated from the approximated sets of concepts, we have removed any further redundancy that may be hidden within the condition parts of decision rules.

**Ease of Access**
As we have proposed a level-wise design of KB, rules with \( k \) facts \((1 \leq k \leq n)\) in condition parts appear at \((k - 1)th\) level of lattice. So, if we want to find a rule with \( k \) facts, we would directly access the \((k - 1)th\) level. Also, from the lattice structure we can find easily those rules which include fact \( f \) in their condition parts. Furthermore, the unique labels assigned to the Lattice_Element nodes are used for their easy identification. As a result, the access to the Lattice_Element nodes becomes quicker.

7. Conclusion

In this paper, we have indicated how rough set based lattice structure for knowledge representation provides foundation for designing a knowledge base for a medical expert system. As a medical expert system should be reliable, we have proposed a new metric for a rule to measure how much trustworthy the rule is. The proposed knowledge representation scheme generates optimal decision rules which may be certain, possible or uncertain. But as reliability is a major concern in medical expert systems, these kind of systems would mainly consider the certain rules. We also have derived some probabilistic properties of the optimal rules. We have illustrated the whole approach using a simple case study from the domain of low back pain. The proposed design has the properties like completeness, consistency, integrity, and non-redundancy. The design also facilitates ease of access of the rules. We plan to use this scheme for the real-world development of the knowledge base for a medical expert system for LBP management in the near future.

**References**

Abu-Nasser, B. (2017). Medical Expert Systems Survey.

Achour, S. L., Dojat, M., Rieux, C., Bierling, P., & Lepage, E. (2001). A UMLS-based knowledge acquisition tool for rule-based clinical decision support system development. *Journal of the American Medical Informatics Association, 8*(4), 351-360.

Atkins, J. S., Kunz, J. C., Shortliffe, E. H., & Fallat, R. J. (1983). PUFF: an expert system for interpretation of pulmonary function data. *Computers and biomedical research, 16*(3), 199-208.

Allegri, M., Montella, S., Salici, F., Valente, A., Marchesini, M., Compagnone, C., ... & Fanelli, G. (2016). Mechanisms of low back pain: a guide for diagnosis and therapy. *F1000Research, 5*.

Arsene, O., Dumitrache, I., & Mihu, I. (2015). Expert system for medicine diagnosis using software agents. *Expert Systems with Applications, 42*(4), 1825-1834.

Ashraf, S., Murty, M. N., & Shevade, S. K. (2006). Rough set based incremental clustering of interval data. *Pattern Recognition Letters, 27*(6), 515-519.

Aswani Kumar, C., & Srinivas, S. (2010). Mining associations in health care data using formal concept analysis and singular value decomposition. *Journal of biological systems, 18*(04), 787-807.

Belohlavek, R. (2008). Introduction to formal concept analysis. *Palacky University, Department of Computer Science, Olomouc, 47.*

Bichindaritz, I., & Marling, C. (2006). Case-based reasoning in the health sciences: What’s next?. *Artificial intelligence in medicine, 36*(2), 127-135.

Bichindaritz, I., Kansu, E., & Sullivan, K. M. (1998, September). Case-based reasoning in care-partner: Gathering evidence for evidence-based medical practice. In *European Workshop on Advances in Case-Based Reasoning* (pp. 334-345). Springer, Berlin, Heidelberg.

Błaszczyński, J., & Słowiński, R. (2003). Incremental induction of decision rules from dominance-based rough
Appendices

Appendix A

/*The procedure Create_Level_Node is described below*/

Procedure Create_Level_Node (prev, level) /* prev is the Lattice_Level node at the immediate predecessor level
and level is the current level at which new Lattice_Level node is to be created */

/* Allocating space for a Lattice_Level node */
lv = Get_Level_Node();
/* Assigning values to the elements belonging to the structure of level node */
next_level[prev] = lv;
next_level[lv] = NIL;
level[lv] = level;
total_node_level[lv] = nClevel;
arr[lv] = Size_of ((Lattice_Element node) * total_node_level[lv]) /* Allocation of total space for the array of lv */

For i = 0 to total_node_level[lv]−1 Do
    arr[lv][i] = NIL;
Endfor
/* Returning the address of the newly created level node to the Build_KB procedure */
Return lv;
End Create_Level_Node

/* The procedure Create_Lattice_Node is described below. */

Procedure Create_Lattice_Node(lv) /* lv is the Lattice_Level node where a new Lattice_Element node is to be created */

/* Allocating space for a Lattice_Element node */
ln = Get_Element_Node();
/* Assigning values to the structural components of a Lattice_Element node */
label[ln] = Assign_Label(lv, ln); /* The procedure Assign_Label assigns a unique label to ln */
condition[ln] = Assign_Condition(ln, lv); /* Assign_Condition procedure assigns condition to ln */
decision[ln] = Assign_Decision(ln, lv); /* Assign_Decision procedure assigns decision to ln */
pred_successor[ln] = NIL;
successor[ln] = NIL;
/* Returning the address of the newly created Lattice_Element node to the Build_KB procedure */
Return ln;
End Create_Lattice_Node

/* The procedure Create_Predecessor_List is described below. */

Procedure Create_Predecessor_List(ln, level) /* ln is a Lattice_Element node and level represents the current lattice level */

/* Initialization of pointers */
h = NIL, temp = NIL, pn = NIL; /* h is the starting node of the predecessor list of ln, temp is a temporary Node pointer, and pn is a Node pointer to hold newly generated node. */
/* Space allocation for the initial node of predecessor list of ln */
If level == 0 Then
    h = NIL;
Else
    h = Get_Node(); //The procedure allocates space for the h node
    label[h] = 0;
    ptr[h] = NIL;
Endif
/* Generation of consecutive predecessor nodes */
temp = h
For i = 1 to level Do
    pn = Get_Node;
    label[pn] = Predecessor_Label(label[ln], i);
    ptr[pn] = NIL;
    ptr[temp] = pn;
    temp = ptr[temp];
The procedure `Predecessor_Label(label, j)` searches for j-th 1 in the bit string of ‘label’ starting from the leftmost position. Suppose the location of j-th 1 in the bit string is k. The procedure assigns 0 at the k-th position of the bit string and returns the modified bit string.

The procedure `Create_Predecessor_List` is described below.

```
Procedure Create_Predecessor_List(ln, level)
    /* ln is a Lattice_Element node and level represents the current lattice level */
    /* Initialization of pointers */
    h = NIL, temp = NIL, sn = NIL;
    /* h is the starting node of the predecessor list of ln, temp is a temporary Node pointer, and sn is a Node pointer to hold newly generated node. */
    /* Space allocation for the initial node of successor list of ln */
    If level = n Do
        h = NIL;
        Endif
    /* Generation of consecutive successor nodes */
    If level < n Then
        h = Get_Node();
        label[h] = NIL;
        ptr[h] = NIL;
        temp = h;
        For i = 1 to (n – level)
            sn = Get_Node();
            label[sn] = Successor_Label(label[ln], i);
            ptr[sn] = NIL;
            ptr[temp] = sn;
            temp = ptr[temp];
        Endfor
    Endif
    /* Returning the address of the first node of the successor list of ln to the Build_KB procedure */
    Return h;
End Create_Predecessor_List
```

The procedure `Successor_Label(label, j)` searches for j-th 0 in the bit string of ‘label’ starting from the rightmost position. Suppose the location of j-th 0 in the bit string is k. The procedure assigns 1 at the k-th position of the bit string and returns the modified bit string.

Appendix B

/*The following procedure is for approximation generation from modified knowledge when decision d is also included in some other Lattice_Element nodes in KB*/

```
Procedure Modification_Approximation_Generation1(ArrD, NewD) /* NewD is the set of Lattice_Element nodes which now include d in their decision parts */
    /* Initialization of variables */
    count = Size_of(NewD);
    /* Generation of modified approximated sets for Concept1. Note that Concept1 has been modified accordingly */
    l1 = |Upper_Approximation1|, l2 = |Lower_Approximation1|, l3 = |Boundary_Region1|;
    For i = 0 to count – 1 Do
        If V_d = 1 OR V_d = 2 Then
            Upper_Approximation1[i] = Upper_Approximation1 ∪ {NewD_i};
        Endif
    Endfor
```

Modification_Approximation_Generation

/* The following procedure is for approximation generation from modified knowledge when decision $d$ is deleted from some existing $Lattice_Element$ nodes in KB. */

Procedure Modification_Approximation_Generation ($ArrD, OldD$) /* OldD is the set of $Lattice_Element$ nodes from which decisions having disease $d$ are to be deleted */

/* Initialization of variables */

$count = Size_{of}(OldD);$  // $count$ is a temporary variable to hold the size of $OldD$

/* Generation of modified approximated sets for Concept1. Note that Concept1 has been modified accordingly */

For $i = 0$ to $count - 1$ Do

If $V_d == 1$ OR $V_d == 2$ Then

$Upper_{Approximation}_1[l_1] = Upper_{Approximation}_1 - \{OldD\};$

$l_1 \leftarrow l_1 - 1;$

Endif

If $V_d == 1$ Then

$Lower_{Approximation}_1[l_2] = Lower_{Approximation}_1 - \{OldD\};$

$l_2 \leftarrow l_2 - 1;$

Endif

If $V_d == 2$ Then

$Boundary_{Region}_1[l_3] = Boundary_{Region}_1 - \{OldD\};$

$l_3 \leftarrow l_3 - 1;$

Endif

Endfor

/* Generation of modified approximated sets for Concept2. Note that Concept2 has been modified accordingly */

For $i = 0$ to $count - 1$ Do

If $V_d == 1$ Then

$Lower_{Approximation}_2[l_1] = Lower_{Approximation}_1 \cup \{NewD\};$

$l_1 \leftarrow l_1 + 1;$

Endif

If $V_d == 2$ Then

$Boundary_{Region}_2[l_3] = Boundary_{Region}_1 \cup \{NewD\};$

$l_3 \leftarrow l_3 + 1;$

Endif

Endfor

End Modification_Approximation_Generation
If $V_d = 0$ OR $V_d = 2$ Then
  \[ Upper_{\text{Approximation}}_2[l_1] = Upper_{\text{Approximation}}_2 - \{OldD\}; \]
  \[ l_1 \leftarrow l_1 - 1; \]
Endif
If $V_d = 0$ Then
  \[ Lower_{\text{Approximation}}_2[l_2] = Lower_{\text{Approximation}}_2 - \{OldD\}; \]
  \[ l_2 = l_2 - 1; \]
Endif
If $V_d = 2$ Then
  \[ Boundary_{\text{Region}}_2[l_3] = Boundary_{\text{Region}}_2 - \{OldD\}; \]
  \[ l_3 = l_3 - 1; \]
Endif
End Modification_Approximation_Generation_2

/* The following procedure is for approximation generation from modified knowledge when $V_d$ for decision $d$ in a particular Lattice_Element node in KB has been changed. */

Procedure Modification_Approximation_Generation_3($l_n, d$) /*$l_n$ is the Lattice_Element node whose $V_d$ has been changed to $V_{d, new}$ */
If $V_d = 0$ AND $V_{d, new} = 1$ Then
  \[ Upper_{\text{Approximation}}_1 = Upper_{\text{Approximation}}_1 \cup \{l_n\}; \]
  \[ Lower_{\text{Approximation}}_1 = Lower_{\text{Approximation}}_1 \cup \{l_n\}; \]
  \[ Boundary_{\text{Region}}_1 = Boundary_{\text{Region}}_1 \cup \{l_n\}; \]
  \[ Upper_{\text{Approximation}}_2 = Upper_{\text{Approximation}}_2 - \{l_n\}; \]
  \[ Lower_{\text{Approximation}}_2 = Lower_{\text{Approximation}}_2 - \{l_n\}; \]
  \[ Boundary_{\text{Region}}_2 = Boundary_{\text{Region}}_2 - \{l_n\}; \]
Elseif $V_d = 0$ AND $V_{d, new} = 2$ Then
  \[ Upper_{\text{Approximation}}_1 = Upper_{\text{Approximation}}_1 \cup \{l_n\}; \]
  \[ Lower_{\text{Approximation}}_1 = Lower_{\text{Approximation}}_1 \cup \{l_n\}; \]
  \[ Boundary_{\text{Region}}_1 = Boundary_{\text{Region}}_1 \cup \{l_n\}; \]
  \[ Upper_{\text{Approximation}}_2 = Upper_{\text{Approximation}}_2 - \{l_n\}; \]
  \[ Lower_{\text{Approximation}}_2 = Lower_{\text{Approximation}}_2 - \{l_n\}; \]
  \[ Boundary_{\text{Region}}_2 = Boundary_{\text{Region}}_2 - \{l_n\}; \]
Elseif $V_d = 1$ AND $V_{d, new} = 0$ Then
  \[ Upper_{\text{Approximation}}_1 = Upper_{\text{Approximation}}_1 - \{l_n\}; \]
  \[ Lower_{\text{Approximation}}_1 = Lower_{\text{Approximation}}_1 - \{l_n\}; \]
  \[ Boundary_{\text{Region}}_1 = Boundary_{\text{Region}}_1 - \{l_n\}; \]
  \[ Upper_{\text{Approximation}}_2 = Upper_{\text{Approximation}}_2 \cup \{l_n\}; \]
  \[ Lower_{\text{Approximation}}_2 = Lower_{\text{Approximation}}_2 \cup \{l_n\}; \]
  \[ Boundary_{\text{Region}}_2 = Boundary_{\text{Region}}_2 \cup \{l_n\}; \]
Elseif $V_d = 1$ AND $V_{d, new} = 2$ Then
  \[ Lower_{\text{Approximation}}_1 = Lower_{\text{Approximation}}_1 - \{l_n\}; \]
  \[ Boundary_{\text{Region}}_1 = Boundary_{\text{Region}}_1 \cup \{l_n\}; \]
Elseif $V_d = 2$ AND $V_{d, new} = 0$ Then
  \[ Upper_{\text{Approximation}}_1 = Upper_{\text{Approximation}}_1 - \{l_n\}; \]
  \[ Boundary_{\text{Region}}_1 = Boundary_{\text{Region}}_1 - \{l_n\}; \]
  \[ Upper_{\text{Approximation}}_2 = Upper_{\text{Approximation}}_2 \cup \{l_n\}; \]
  \[ Lower_{\text{Approximation}}_2 = Lower_{\text{Approximation}}_2 \cup \{l_n\}; \]
  \[ Boundary_{\text{Region}}_2 = Boundary_{\text{Region}}_2 \cup \{l_n\}; \]
Elseif $V_d = 2$ AND $V_{d, new} = 1$ Then
  \[ Lower_{\text{Approximation}}_1 = Lower_{\text{Approximation}}_1 \cup \{l_n\}; \]
  \[ Boundary_{\text{Region}}_1 = Boundary_{\text{Region}}_1 - \{l_n\}; \]
Endif
End Modification_Approximation_Generation_3
