Available online at www.elixirpublishers.com (Elixir International Journal)

Information Technology

Elixir Inform. Tech. 103 (2017) 45705-45711

ISSN: 2229-712X

Frequent Case Tree Algorithm to support patient case indexing in a handheld Clinical Decision Support system

P V K Sandeep
Electronics and Computer Engg (ECM) Department, SreeNidhi Institute of Science and Technology, Hyderabad, Telangana, India.

ARTICLE INFO

Article history:
Received: 2 January 2017;
Received in revised form:
4 February 2017;
Accepted: 14 February 2017;

Keywords
Data mining,
Clinical Decision Support Systems,
Frequent Case Tree,
Rule Based indexing,
Medical informatics,
Hand-held devices.

ABSTRACT
In modern world, decision support systems are gaining increased popularity in various domains, including medical informatics and engineering. Digital access to the patient case data in would help the novice nurses in making faster and accurate decisions. The discussions in this paper are based on N-CODES (Nursing Computer Decision Support), an interactive handheld device which takes in patient data and delivers clinical knowledge i.e. it provides a diagnostic for decision support by the filtered and collated data. The focus of this paper is mainly on one aspect, the indexing of patient specific cases, and individual rule fragments. 'Improve' data library has been used, which contains instances of each possible patient class in the clinical decision support system (CDS). These classes are organized into frequent case (FC) trees by developing an algorithm which will find the frequently occurring rules. This algorithm will support the patient case indexing by organizing cases into rule vectors consisting of rule instances for a specific case ending with an intervention and also will aid in determining similarity of new cases with stored exemplar cases.

Introduction

In the medical domain, we see that clinics and hospitals collect a large amount of patient data. To access this data and to make decisions based on this, the novice nurses would definitely require some guidance from the senior nurses. This is fairly due to the inexperience of the novice nurses and also high patient loads. Hence it would be ideal to have a decision support system which makes faster and accurate decisions. Also the novice nurses can have digital access to all the real-time patient data.

The promise of clinical decision support (CDS) systems is to provide tools for physicians, nurses, staff, patients or other stake holders with general knowledge and patient specific information, intelligently collected, filtered, collated, and presented in a timely fashion to enhance health, safety and patient outcomes. Such CDS systems would include a variety of computer supported tools for data collection, data fusion, condition alerts and reminders, clinical guidelines, care order sets, patient data reports, diagnostic support and clinical workflow assistants to name a few. Some targeted CDS systems have been effective in improving outcomes and reducing errors at some healthcare institutions by making needed medical knowledge readily available to knowledgeable healthcare providers. At the same time CDS systems have proven quite problematic and have not therefore become readily available at most sites.

The discussions in this paper are focused on N-CODES (Nursing Computer Decision Support), a project done at the University of Massachusetts Dartmouth by nurses and engineers together to develop a hand-held device to aid the Clinical decision support. The results of this paper are achieved by using the 'Improve' data library, a database built for the N-CODES.

Tele: +91 – 800 888 5484
E-mail address: sandeep5484@gmail.com

This paper also discusses in detail about the patient case indexing and frequent case tree algorithm to find frequently occurring patient cases.

Overview of N-CODES

As the patient acuity increases and cost-effective treatment becomes mandatory, the novice nurses often end up in a perplexed situation wherein they must take complex decisions almost on regular basis [6]. N-CODES (Nursing Computer Decision Support System), project at the University of Massachusetts Dartmouth addresses this problem by developing a prototype to provide some clinical knowledge through an interactive device using wireless access. N-CODES is an intelligent handheld device which takes in patient information and provides a diagnostic for decision support after evaluating the patient's condition using the given information. The architecture of the N-CODES system can be seen in Figure 1.

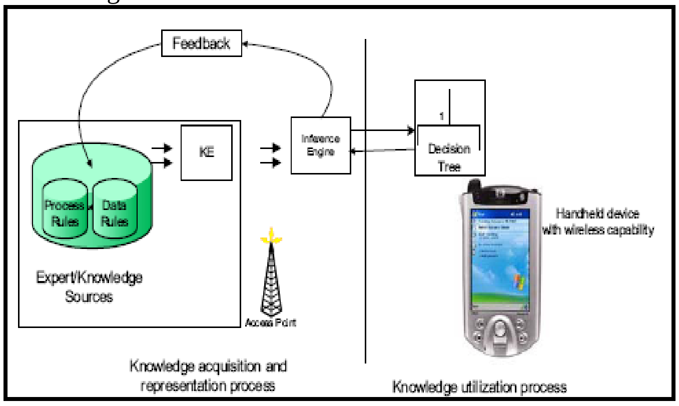

Feedback
KE
Inference Engine
Decision Tree
Process Rules
Data Rules
Expert/Knowledge Sources
Access Point
Handheld device with wireless capability
Knowledge acquisition and representation process
Knowledge utilization process

Figure 1. NCODES Architecture [4].

Three main functions of the handheld device are [1]:
• Enabling the patient information to be downloaded from the server
• Enabling continuous use of patient information stored on the PDA
• Collecting extra information such as vital signs and other assessments

After all the information is collected, it is then used to assist the nurse in proper evaluation and making appropriate interventions with the help of the previous and existing cases. NCODES uses the guidance of an expert rule-based engine to provide intelligent decision support. It also uses the clinical information as a knowledge base to guide through the patient treatment process. The system provides the user with some advice and also the reason behind this advice.

The inference engine ensures that the nurse can transit between states to browse through the information given to the system. Using the given inputs, process rules and the case history, the inference process computes the next possible valid state and gives an output which is reliable for the patient assessment.

The system's decision making capability is realized by the traversal of the decision tree from baseline assessment to pain assessment to cough assessment and more specifically to the refined assessment. This is of course dependent upon the rules existing in the database. The framework follows rule based reasoning considering logical rules as basic modules and the production rules as relationships amongst rule clusters [1]. NCODES pronounces these as rules and process rules.

Rule-based Systems

A rule-based system (RBS) consists of a rule-base (permanent data), an inference engine (process), and a workspace or working memory (temporary data). The RBS uses facts to represent knowledge. Facts are those which consist of rules in logic (for reasoning) and production rules (working memory + rule base). The production rules express the knowledge of a human expert, thus making RBS to solve similar problems on its own. A rule base typically consists of rules which contribute to the domain knowledge [6].

The RBS compares facts about a new problem that are stored in a knowledge base against a stored set of rules. When there is a match, the rule will fire. This rule can be enabled in such a manner that it can fire another rule thus creating a chaining effect. The process continues until there are no more rules to fire and the output is presented to the user. The advantage with RBS is the simplicity of input variables. In general a rule is supposed to be of the format

If <condition clause> Then <action clause>

The condition clause checks the state of the temporary data. If the condition is satisfied then the consequent set of actions are triggered by the action clause.

Case-Based Reasoning

Case-Based Reasoning (CBR) is a problem solving technology which looks for a similar problem in a case-base and retrieves the solution of this past case. The retrieved solution is then used as a starting point to find the solution of the actual problem. CBR can be described as a cyclic process comprising of:

- Case Retrieval: Retrieving most similar case
- Case Reuse: To attempt to solve the problem
- Revising proposed solution
- Retaining the new solution as part of a new case

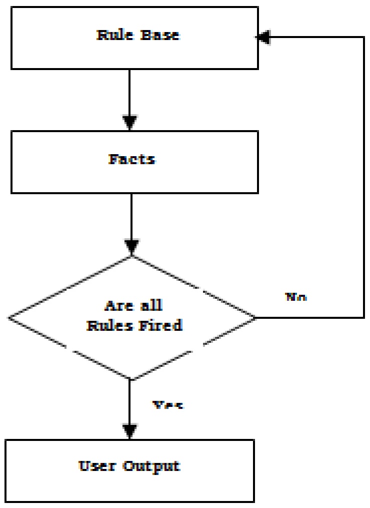


Figure 2. Rule-Based System [6].

To understand CBR, we need to know something about problem and solution. Both, problems and solutions are represented in some data structure as attribute-value vectors. Certain variables, instantiated for an actual situation may occur in problem as well as solution but it usually important for the problem to find appropriate variable for the description of the solution [9].

A case is a contextualized piece of knowledge representing an experience [10]. The basic idea of a case is to record an episode where a problem is solved. This includes partially solved problems too. A case is represented as an ordered pair (problem, solution) [9] which can be further refined to say that a case is a collection of rules. A set of cases collected together for retrieval purposes is termed as a Case Base. A case base represents some class of past experience.

Indexing

In the modern world, enormous amount of data is being collected in most of the domains including medical, engineering, military and business. All the data collected is stored in a database. As the size of a database increases, data retrieval becomes tougher because it takes agonizingly large period of time. In order to avoid this, most database systems use indexes to speed up the retrieval of data. An Index is a computational data structure which could be kept in the memory for a faster searching. This makes the computer to avoid searching each and every record stored in the database. Indexes are employed even in CBR to speed up the retrieval process. When employing an index in a CBR, it should [10]:

• Be predictive
• Address the purposes of the case
• Be abstract enough to allow for widening the future use of case-base
• Be concrete enough to be recognized in the future

The information in a case is stored as indexed and un-indexed information. The method used to select an index can be either manual or automatic. Choosing indexes manually involves deciding a case's purpose with respect to the aims of the system and deciding under what circumstances the case will be useful. For practical applications, indexes can be chosen automatically, manually or by both techniques but most of the CBR tools presently on the market are seen supporting automatic identification of case indexes [10]. One aspect of this paper is to propose an algorithm to build indexes. Indexes are built for the frequent rules and frequent cases which have been extracted from the existing patient database known as the Improve Data Library.

The ‘Improve’ Data Library will be discussed later in detail. Also, how the frequent rules and frequent cases are extracted and how they are further used to construct the frequent case tree will be discussed in next sections.

Frequent Case Tree (FC Tree)

This section discusses the basic concepts of a Frequent Case Tree (FC Tree). An FC tree is basically a compressed representation of the input data. It is a tree structure which consists of nodes. Each node has the item name, the frequency count and the node link. An FC tree is similar to an n-ary tree where a node is linked to any number of nodes. The construction of an FC tree involves reading a case from the case index and mapping onto a path in the FC tree. Since different cases can have several rules in common, the paths may overlap. More number of overlaps implies more compression in the FC-tree structure. This makes it to use less memory and fit easily into the database thus allowing us to extract frequent items directly from the structure in database.

In the current paper, case frequency is used to point into the forest of trees the most frequently accessed items and over time provide information on possible mutations to the FC tree to examine and changing the composition of cases in FC tree. This paper shows an algorithm sophisticated enough to extract the frequent items and map them onto a path in the FC tree. The procedure involved in the algorithm will be discussed in detail in a later section.

Consider an ontology base which consists of all the cases that have occurred to date. Each case in ontology is an individual representation of a disease related group (DRG). There is every possibility that a set of rules in one case can match with a set of rules in some other case. This can be proved using the hit ratio algorithm which will be discussed in later section. These similar cases can be prioritized so that nurses can refer to these cases to draw conclusions. This method is known as *Best Evidence Case Practice*. Thus prioritized outcomes can be sent to the inference engine. When building the initial FC tree using the ontology, ‘1’ leaf (case) would be present for each case of ontology. Thus N leafs would be present.

‘Improve’ Data Library

The dataset used for publishing the results of this paper has been taken from the *“Improve Data Library”*. *IMPROVE (Improving Control of Patient Status in Critical Care)* project began in October 1994 with a goal to improve the on-line assessment and management of the patient state in critical care and also to improve the operating theatre environments by applying innovative nonlinear, linear, multivariate and knowledge based algorithms [5]. The data library was obtained in the second phase of the project and the actual collection and annotation of the data was carried out at the Department of Intensive Care of Kuopio University Hospital (KUH) at Finland. The project ended in January 1997 with the accomplishments of two tasks i.e. collection of patient data and utilization of this data library for the development of bio-signal interpretation methods. The dataset contains the recordings of 59 patients which were obtained from episodes of roughly 24 hours of observation. For 7 of these patients, their EEG has also been recorded in addition to the current recordings. Out of these 59, some of them were admitted more than once but each admission was considered as a separate recording. Out of these 59 patients, 7 were admitted twice and 1 was admitted thrice [1].

To ensure quality in the measured data and also to check for any missing annotations, video monitoring has been done. All the data is collected by the Improve data collection system and the recorded signals were stored in some temporary files on the hard disk. The CIMS (Clinical information management system) records all the data from the annotations, patient monitors, nursing actions and the lab results and stores in the CIMS database [1].

Disorders Included in the Data Library

As part of the ‘Improve’ project, a survey group known as the clinical task group was set up. One of the objectives of this group was *to define the clinical disorders*. The other objectives include supervision of the data collection, conducting quality control of the annotation process and to do the interim and final analysis of the library. After conducting a survey in three ICUs, they were able to define four categories of disorders. Hypovolaemia, Cardiac Failure, High Flow State and O_2 content related problems are the four major disorders which have been included in the data library. Each main group was again divided into subgroups. According to the disorders in ICU, a patient can be classified into one or more of these four disorders [1]. In this section we discuss the criteria required to classify the patient into a particular disorder.

Hypovolaemia

1. Inadequate venous return, low CVP and signs of insufficient flow [5]

☞ Primary criteria [5]:

- Low filling pressure in relation to the patient’s cardiovascular performance
- Low cardiac index < 2.0
- Low peripheral temperature, $T_p < 32.5$ °C excluding local vascular disorders

☞ Secondary criteria [5]:

- Metabolic signs of tissue hypoxia ($\approx$ 1 - 2 h) & metabolic acidosis
- pH < 7.35 & BE < – 4
- Lactataemia > 2 mmol/L
- O2 consumption, SvO2 $\leq$ 65 %, extraction ratio > 0.40

2. Adequate venous return and no signs of insufficient flow [5]

☞ Criteria [5]:

- Adequate PCWP $\geq$ 6 and MAP $\geq$ 65 mmHg
- Continuous volume replacement > 1000 ml/h or need of episodical volume replacement with higher infusion rate.

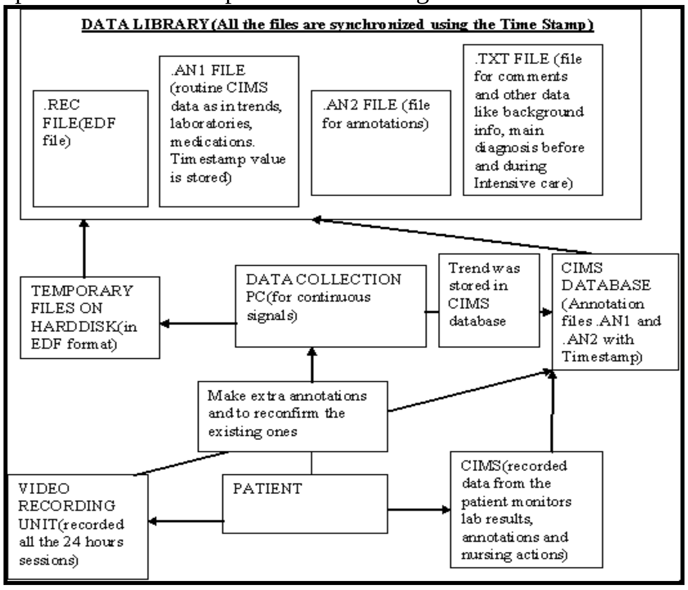

DATA LIBRARY (All the files are synchronized using the Time Stamp)
.REC FILE(EDF file)
.AN1 FILE (routine CIMS data as in trends, laboratories, medications. Timestamp value is stored)
.AN2 FILE (file for annotations)
.TXT FILE (file for comments and other data like background info, main diagnosis before and during Intensive care)
TEMPORARY FILES ON HARDDISK(in EDF format)
DATA COLLECTION PC(for continuous signals)
Trend was stored in CIMS database
CIMS DATABASE (Annotation files .AN1 and .AN2 with Timestamp)
Make extra annotations and to reconfirm the existing ones
VIDEO RECORDING UNIT(recorded all the 24 hours sessions)
PATIENT
CIMS(recorded data from the patient monitors lab results, annotations and nursing actions)

Figure 3. Improve Data Library Collection System [1].

Cardiac failure

1. Inadequate flow and metabolic signs of tissue hypoxia [5]

☞ Primary criteria [5]:

- Low CI and high /normal filling pressure (CI < 2.0 ; PCWP >10)
- Metabolic signs of tissue hypoxia as above

☞ Secondary criteria [5]:
- Low urine output (optional) ≤ 0.5 – 1.0 ml/kg/h, over 2h
- Low Tp

2. Inadequate flow, no signs of tissue hypoxia [5]

☞ Criteria [5]:
- Low CI and high/normal filling pressures PCWP > 10 (primary criteria)
- Low urine output (optional)
- Low Tp

3. Acceptable flow and continuous need of exceptional support [5]

☞ Primary criteria [5]:
- Low/normal CI, CI > 2.0, need for high filling pressures and PCWP >10
- Low/normal CI and normal/high filling pressure + need of vasodilatations
- Need of inotropic drugs

Cardiac failure is defined separately for left and right ventricular failures based on clinical judgment.

High flow state

1. Abnormally high flow and need of vasoactive treatment to maintain perfusion pressure (dopamine, norepinephrine) for low SVRi, SVRi < 120 [5]

☞ Primary criteria [5]:
- High CI > 4.0 and dopamine or norepinephrine
- Tp normal or high Tp > 32.5 °c

2. Abnormally high flow, acceptable perfusion pressure and signs of tissue hypoxia [5]

☞ Criteria [5]:
- High CI & Metabolic signs of tissue hypoxia
- Need to increase DO2 by inotropic drugs
- Low O2 extraction
- O2 consumption, SvO2 ≤ 65 %, extraction ratio > 0.40
- Tp normal or high

3. Abnormal high flow & acceptable perfusion pressure [5]

☞ Primary criteria [5]:
- High CI, 4.0
- Tp normal or high ≥ 32.5
- Low O2 extraction

Oxygen content related problems

1. Desaturation [5]

☞ Primary criteria [5]:
- Low SaO2 < 90 and oxygen mask
- High respiratory frequency and/or manifestation of efforts and distress
- Metabolic signs of tissue hypoxia

2. Acute ventilatory failure and Desaturation [5]

☞ Criteria [5]:
- Low pH ≤ 7.35 and high PaCO2 ≥ 6.0 kPa
- Low SaO2 < 90

3. Desaturation, signs of tissue hypoxia and need for maximum ventilatory support [5]

☞ Primary criteria [5]:
- Low SaO2
- Low High $\frac{PaO_2}{FiO_2}$ < 150 mmHg
- Metabolic signs of tissue hypoxia

☞ Secondary criteria [5]:

Need of inotropic drugs (to increase O2 delivery)

4. Normal content with maximum ventilatory support [5]

☞ Criteria [5]:
- Normal SaO2 > 90
- $\frac{PaO_2}{FiO_2}$ < 200 mmHg
- No signs of tissue hypoxia or need of inotropic drugs

5. Normal content with routine ventilatory support [5]

☞ Criteria [5]:
- Normal/low SaO2 , SaO2 < 90
- Normal/high PaCO2, PaCO2 > 4.5 kPa
- No signs of tissue hypoxia or need of inotropic drugs

Table 1 lists some of the abbreviations used while describing the disorders.

Table 1. List of Abbreviations used to describe disorders. [5]

Abbreviation	Description
MAP	Mean arterial pressure
CVP	Central venous pressure
T_p	Peripheral temperature
HR	Heart rate
ResF	Respiratory frequency
CI	Cardiac index
PCWP	Pulmonary capillary wedge pressure
SVR_i	Systemic vascular resistance index
PaO2	Arterial oxygen tension
FiO2	Fraction of inspired oxygen
aB-pH	Arterial pH
DO2, VO2	Oxygen delivery, oxygen consumption
ExO2	Oxygen extraction
aB-BE	Base excess
Lact	Blood lactate concentration

Process Rules and Decision Trees

To develop an algorithm, it is important that we have some use cases for each patient episode. But in order to build a use case for a patient episode it is required that we have a decision tree and also develop some process rules. Decision tree is a powerful and popular tool used for classification and prediction. In general it is considered as a data structure or rather a tree structure used in particular for classification tasks, where each node is either a leaf node or a decision node. A decision tree can be used to classify an example by starting at the root of the tree and moving through until it reaches a leaf node. The leaves represent classes whereas each non-leaf node represents an attribute whose value is used as a label for an edge starting from a node.

A path in the decision tree represents an implication saying that an object with specific feature values belongs to a certain class. A decision tree is considered advantageous in that it represents *rules* which can readily be expressed and understood by a common man i.e. a decision tree can be transformed into *process rules*. The rules formed from such a tree are of the type *If-Then*. A rule can be created for each path from the root to the leaf node. The attributes in the path form a conjunction for the *If* part. The leaf node which is the outcome of the *If* clause forms the *Then* part. Thus rules can be helpful in classifying given data or predicting outcome of an unknown sample [1].

The other advantages of a decision tree include:
- Simple to understand and interpret
- Able to generate understandable rules
- Able to handle both continuous, categorical variables
- Perform classification without requiring computation
- Provides a clear indication of which fields are most important for prediction or classification
- Robust, performs well with large data in short time

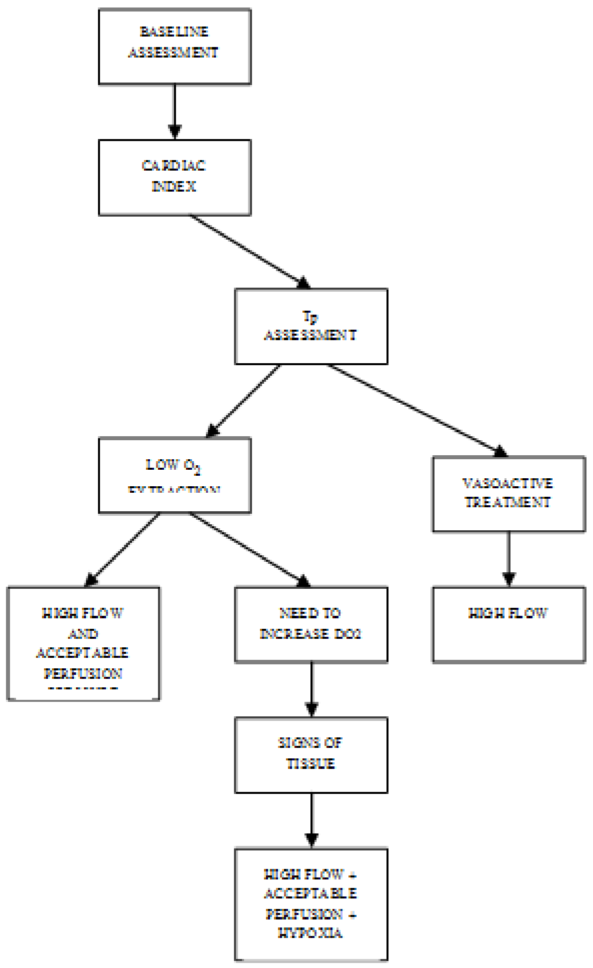


Figure 4. Decision Tree showing only high flow state paths.

Decision Tree for Improve Data Library

The decision tree for the '*Improve*' data library is divided into two parts. One part of the decision tree shows the cardiac index assessment which is the primary criteria in all the disorders except in the oxygen content related problems which is shown in the second part of the tree. The decision tree follows the path from Baseline assessment to an intervention at the leaf node.

The outcome of this intervention can be observed using the patient cases. In between these nodes, the path goes through pain assessment, cough assessment then to refined assessment. The intermediate nodes used to build the decision tree were based on the clinical description of the Improve data library. Thus, two decision trees have been built for the Improve data library. The first part contains the tree for the disorders Hypovolaemia, High flow state and cardiac failure. On the other hand, the second tree is for the O_2 content related problems.

The path in the decision tree, followed by the clinician, for each patient by validating the intervention or leaf node is known as the use case of that particular patient. Both stored procedure method and association rule mining method are used to construct the use cases for 59 patients but the stored procedure method proved more accurate and successful. Out of these, for the patients 1, 4, 16 and 32 only a part of the decision tree was obtained i.e. it ended at the baseline assessment.

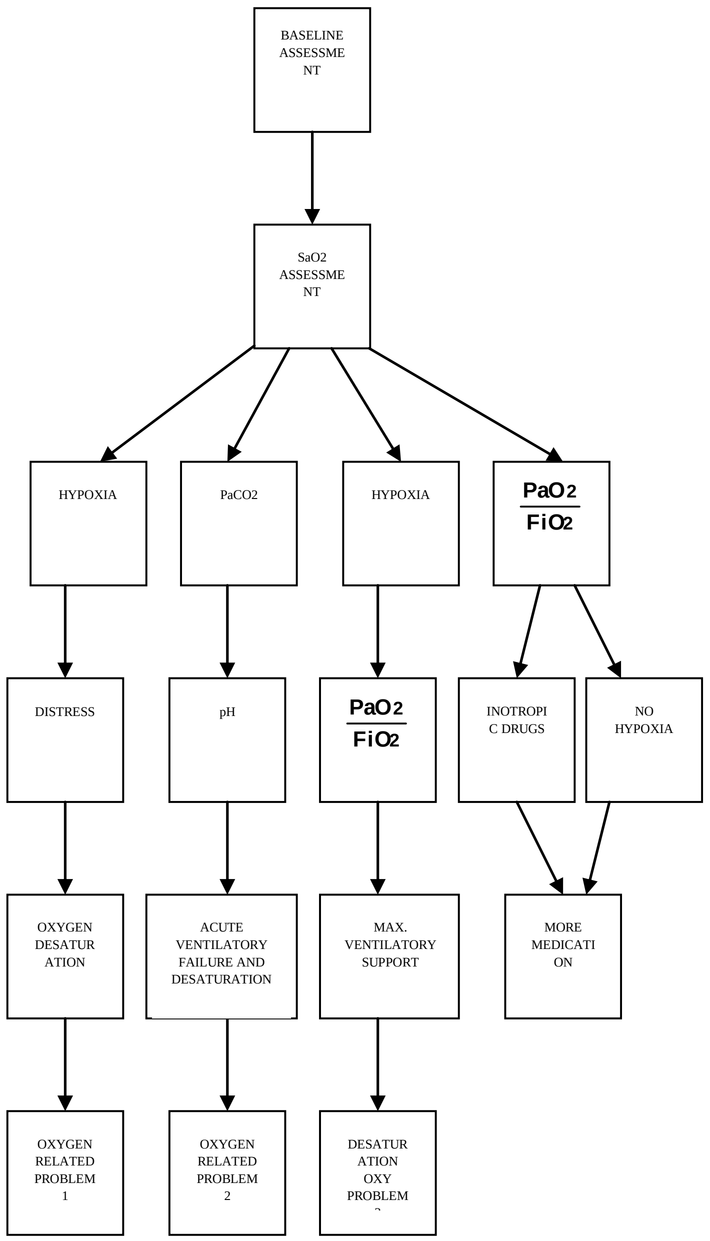


Figure 5. Decision Tree showing Oxygen problems.

Development of Frequent Case (FC) Tree

The first step involved is to build the frequent rule index which contains the frequently occurring rules in the data library. Since the dataset is very large, to search for a rule or a case in this dataset would be a tedious process. To reduce the amount of time for searching, indexing is employed. To make the system work faster and also to improve the efficiency, the frequently occurring rules and cases are identified and an index is built.

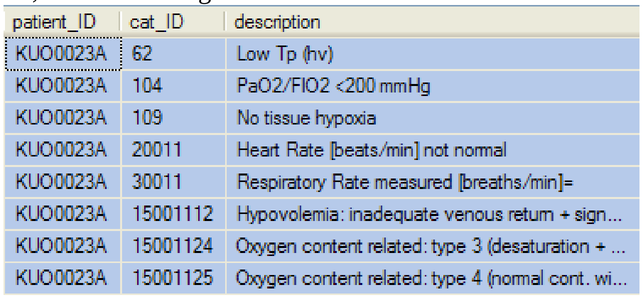
The rule index contains the Rule ID and the rule count. Each rule contains an If ID and a Then ID. With these frequent rules, a frequent rule tree is formed wherein one rule is connected to another till a case i.e. an intervention is reached thus forming a frequent case tree.

Algorithm to develop an FC Tree:

1. Get the patient ID
2. Build the Use Case of this patient
3. Extract all the If_ID from this use case
4. Consider one If_ID and give it an r_id
5. Extract all Then_ID possible with this If_ID from Process_Rules
6. Consider one Then_ID

7. If Then_ID is present in the use case then go to step 6 and proceed
8. Else put this in ignore list and go to step 6 and proceed
9. Repeat step 7 till all the Then_ID are exhausted
10. Go to step 4 and proceed
11. Repeat Step 9 till all the If_ID are exhausted
12. Get all the Rule_ID and the r_id for the respective If_ID
13. Get one Rule_ID
14. If the Rule_ID is already present, add rule count and give reference ID as its previous current_rid
15. Else add the Rule_ID with reference ID as 'NULL'
16. Start with current_ rid and map to the respective child_rid
17. Repeat step 14 till a case or an intervention is found
18. Go to step 12 and repeat process till all the Rule_ID are exhausted
19. Go to Step 1 and repeat process till all the patients are completed. Figure 12 shows all these steps as flowchart.

Now, consider the figure 6 below

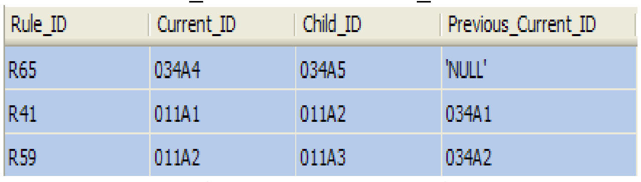


patient_ID	cat_ID	description
KUO0023A	62	Low Tp (hv)
KUO0023A	104	PaO2/FIO2 <200 mmHg
KUO0023A	109	No tissue hypoxia
KUO0023A	20011	Heart Rate [beats/min] not normal
KUO0023A	30011	Respiratory Rate measured [breaths/min]=
KUO0023A	15001112	Hypovolemia: inadequate venous return + sign...
KUO0023A	15001124	Oxygen content related: type 3 (desaturation + ...
KUO0023A	15001125	Oxygen content related: type 4 (normal cont. wi...

Figure 6. Use Case of a Patient 'KUO0023A'.

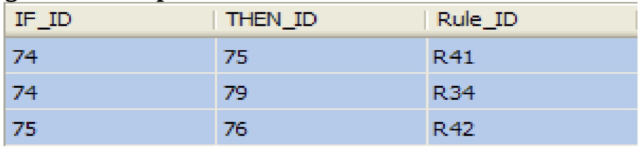
Figure 6 shows the use case of a patient with ID 'KUO0023A'. The If_ID are taken from cat_ID column. Now each ID is given an r_id which is useful in determining node ID of the FC tree.

For example:	If_ID = 62 ➔	r_id = 0023A1
	If_ID = 104 ➔	r_id = 0023A2
	If_ID = 109 ➔	r_id = 0023A3 and so on
	If_ID = 15001125 ➔	r_id = 0023A8



Rule_ID	Current_ID	Child_ID	Previous_Current_ID
R65	034A4	034A5	'NULL'
R41	011A1	011A2	034A1
R59	011A2	011A3	034A2

Figure 7. Excerpt of FC Tree in tabular format in database.



IF_ID	THEN_ID	Rule_ID
74	75	R41
74	79	R34
75	76	R42

Figure 8. Excerpt of Rule_ID from Process_Rules table.

Results

The 'Improve' data library was imported into Microsoft SQL Server 2005 which has been used as the backend database and all the algorithms have been developed in Visual Studio using C# as the programming language. Several PL/ SQL functions and procedures have been written to access the database and also to perform operations on the data structures which have been created. All this has been done on *'improve_all'* database which is an extension to the *'improve'* database.

Results for Rule Index

The first step involved in the construction of the FC tree is to build a rule index. Since the decision tree has been rebuilt, new use cases were found. As a result, new rules have been added to process rules table as shown in Figure 9.

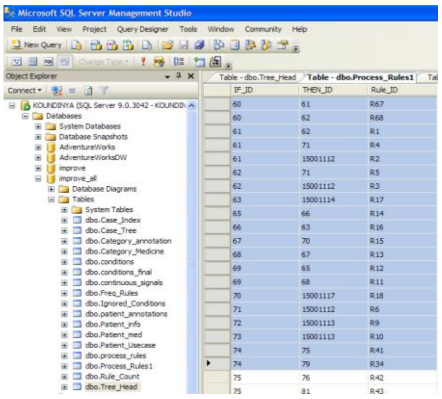


Table - dbo.Process_Rules1

IF_ID	THEN_ID	Rule_ID
60	61	R67
60	62	R68
61	62	R1
61	71	R4
61	15001112	R2
62	71	R5
62	15001112	R3
63	15001114	R17
65	66	R14
66	63	R16
67	70	R15
68	67	R13
69	65	R12
69	68	R11
70	15001117	R18
71	15001112	R6
72	15001113	R9
73	15001113	R10
74	75	R41
74	79	R34
75	76	R42
75	81	R43

Figure 9. Rule Index.

Results for Frequent Rule Index

The rule index shows the list of all the rules present in the system. In order to build a frequent case tree, we need to extract the rules which are occurring frequently. An algorithm which would extract the frequently occurring rules in the database has been written in C#. The extracted rules have been created as an index which contains the Rule ID of the frequently occurring rules and also how frequently the rule is occurring.

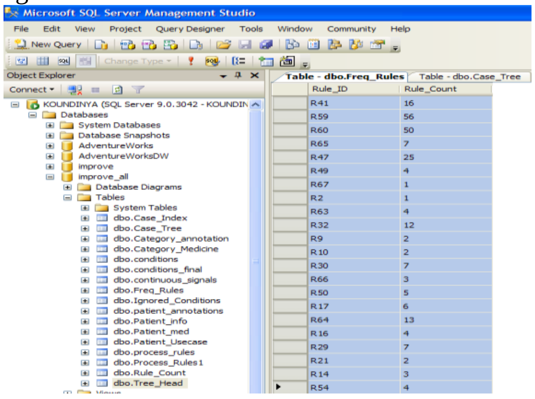


Table - dbo.Freq_Rules

Rule_ID	Rule_Count
R41	16
R59	56
R60	50
R65	7
R47	25
R49	4
R67	1
R2	1
R63	4
R32	12
R9	2
R10	2
R30	7
R66	3
R50	5
R17	6
R64	13
R16	4
R29	7
R21	2
R14	3
R54	4

Figure 10. Frequent Rule Index.

Results for Frequent Case Tree

The next step is to build the frequent case tree. All the frequent rules which have been obtained are mapped onto the FC tree. Each rule is traced down the path of the tree till an intervention or a case is found. This is done for all the patients thus forming many paths in the tree. An algorithm has been written in C# to map all frequent rules onto the FC tree till the respective case or intervention is reached. Figure 11 shows the frequent case tree that has been achieved after implementing the algorithm mentioned earlier.

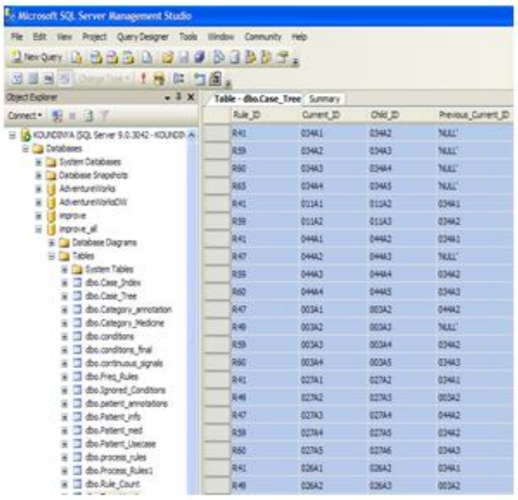


Table - dbo.Case_Tree

Rule_ID	Current_ID	Child_ID	Previous_Current_ID
R41	034A1	034A2	'NULL'
R59	034A2	034A3	'NULL'
R60	034A3	034A4	'NULL'
R65	034A4	034A5	'NULL'
R41	011A1	011A2	034A1
R59	011A2	011A3	034A2
R41	044A1	044A2	034A1
R47	044A2	044A3	'NULL'
R59	044A3	044A4	034A2
R60	044A4	044A5	034A3
R47	003A1	003A2	044A2
R49	003A2	003A3	'NULL'
R59	003A3	003A4	034A2
R60	003A4	003A5	034A3
R41	027A1	027A2	034A1
R49	027A2	027A3	003A2
R47	027A3	027A4	044A2
R59	027A4	027A5	034A2
R60	027A5	027A6	034A3
R41	026A1	026A2	034A1
R49	026A2	026A3	003A2

Figure 11: Frequent Case Tree

Flowchart for developing FC Tree

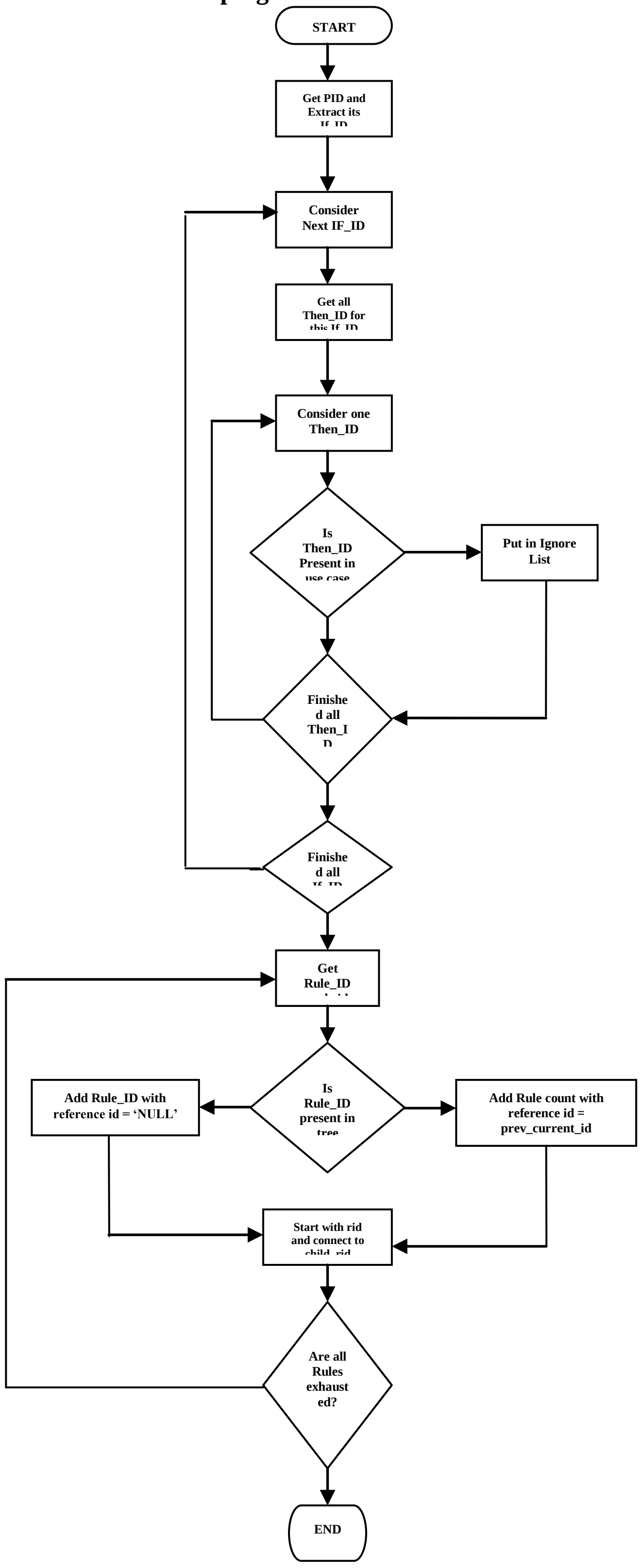

START
Get PID and Extract its IF_ID
Consider Next IF_ID
Get all Then_ID for this IF_ID
Consider one Then_ID
Is Then_ID Present in use case
Put in Ignore List
Finished all Then_ID
Finished all IF_ID
Get Rule_ID
Is Rule_ID present in tree
Add Rule_ID with reference id = 'NULL'
Add Rule count with reference id = prev_current_id
Start with rid and connect to child_rid
Are all Rules exhausted?
END

Figure 12. Flow Chart showing the Algorithm for FC Tree.

Conclusion

An algorithm has been developed which is sophisticated enough to extract frequent items and then build a frequent rule index for these. With this frequent rule index, FC tree was developed by mapping the frequent rules in the index onto the frequent tree. Constructing a frequent rule index and FC tree helps in making the system faster and thus more efficient. This also helps the nurses to follow the method of best evidence case practice.

Future Scope

The FC tree which has been developed and also the algorithm supporting patient case indexing will aid in determining similarity of new cases with stored exemplar cases i.e. similarity searching. An algorithm can be developed which will calculate the hit ratio of a case i.e. it shows the number of times a case has been referred to in the database.

Also, effort can be made in the future *to build a prioritized index of the rules in the FR-index depending on the medical importance*. This is important because if there are any (rules/ cases) in the top level or any levels which are same, then rule based criticality or disease related group (DRG) can be used to reorder or reorganize the existing rules and cases. Consider two rules which have same priority. These rules can be reordered based on how critical the rule is. In case both the rules are critical then the DRG could be checked. The DRG cannot be same because each case is an individual representation.

Acknowledgment

I profusely thank Dr. Paul J. Fortier, Professor, ECE Department, UMass Dartmouth for involving me in this collaborative, medical-engineering related, database project. I would also like to extend a word of thanks to Dr. Liudong Xing and Dr. Nancy Dluhy for committing their time and giving valuable inputs.

References

[1] Shweta J. Brahme, "Methods to Develop a Use case for patient episode", Master's Thesis, University of Massachusetts Dartmouth, January 2007.
[2] Guanjie Yang, "Development of rule based point-of- care decision support system", Master's Thesis, University of Massachusetts Dartmouth, January 2006.
[3] P. Fortier, S. Jagannathan, H. Michel, N. Dluhy, E. Oneill, "Development of a Hand-held Real-time Decision Support Aid for Critical Care Nursing", Proceedings of 36th Hawaii International conference on system sciences, IEEE 2002.
[4] P.Fortier, B.Sarangarajan, H.Michel, N.Dluhy, E.Oneill, "A Compterized Decision Support Aid for Clinical care Novice nursing", Proceeding 38th Hawaii International Conference on System Sciences, IEEE 2005.
[5] K. Nieminen, R. M. Langford, "A Clinical Description of the IMPROVE Data Library", IEEE Engineering in Medicine and Biology, November/ December 1997.
[6] Beena Sarangarajan, "Developing a Framework for Clinical Decision Support Systems", Master's Thesis Dissertation University of Massachusetts Dartmouth, May 2004.
[7] P.Fortier, H.Michel, N.Dluhy, E.Oneill, "The N-CODES Project - The First Year", CIN, Vol. 22, No. 6, 001-006, November/ December 2004.
[8] "http://www.ecfc.u-net.com/cost/rule.htm"
[9] Mario Lenz, Brigitte Bartsch-Sporl, "Case-Based Reasoning Technology", Springer Publications, 1998.
[10] Ian Watson, "Applying Case-Based Reasoning", Morgan Kaufmann Publishers, 1997.
[11] Pang-Ning Tan, Michael Steinbach, Vipin Kumar, "Introduction to Data Mining", Addison Wesley Publishers, 2006.