Joan Albert López-Vallverdú

Knowledge-Based Incremental Induction of Clinical Algorithms

A thesis submitted for the degree of

PhilosophiæDoctor (PhD)

Advisor: Dr. David Riaño

Departament d'Enginyeria Informàtica i Matemàtiques

Universitat Rovira i Virgili

Tarragona
2012

Abstract

Clinical Algorithms (CAs) are flowcharts that graphically summarize some of the medical procedures described in Clinical Practice Guidelines (CPGs), such as diagnosis, management or treatment of certain diseases. Generally, the development of CAs is done manually with the cooperation of several health care experts of different specialties which represents a laborious task. Moreover, the individual differences of the patients causes great variances in the application of CAs in daily practice. In real world, chronic patients use to suffer from more than one disease (comorbidities) and each case has some particularities that may not be considered by the CA.

The automatic induction of structures like CAs from hospital databases and medical resources solves the previous drawbacks. It reduces the high costs of the manual generation and it allows the analysis of health care in comorbidities. Today some computer technologies exist to carry out the induction of procedural knowledge represented as CAs from the hospital databases, but they suffer from several drawbacks: the structures produced by these technologies may not be explicit medical structures that doctors are familiar or satisfied with, they are only based on statistical measures that do not necessarily respect medical criteria which can be essential to guarantee medical correct structures, or they are not prepared to deal with the incremental arrival of data which is worth to consider in medicine where hospital databases are constantly updated with new information generated during daily practice.

In this thesis, we propose a methodology to automatically induce medically correct CAs from hospital databases. These CAs are represented according to a knowledge model called SDA. The methodology considers

relevant background knowledge of a medical domain that has been previously validated by health care experts, and it is able to work in an incremental way, so that the CAs generated are updated as soon as new data arrive.

The methodology has been tested in the domains of hypertension, diabetes mellitus and the comborbidity of both diseases. As a result, we propose an effective repository of background knowledge for all these pathologies and provide the SDA diagrams which have been automatically induced from hospital databases using this repository. Later analyses show that the results are medically correct and comprehensible when validated with health care professionals, and compared against the results obtained by previous technologies.

To my wife, my family and my friends for boosting me and supporting me each step of the way.

Acknowledgements

I am sincerely grateful to my advisor, David Riaño, for the support and guidance he showed me throughout the development and writing of this thesis. I am sure it would have not been possible without his help. Besides I am truly indebted and thankful to Dr. Antoni Collado for his continuous support leading the group of health-care professionals from the SAGESSA Health-Care Group, and providing continuous medical support which has been essential in this work. Finally, I would like to show my gratitude to all the professors, colleagues and classmates who, in one way or another, helped me to achieve this challenge.

Contents

List of Figures

LIST OF FIGURES

LIST OF FIGURES

List of Tables

List of Acronyms

ATC	Anatomical Therapeutic Chemical
BMI	Body Mass Index
BP	Blood Pressure
CA	Clinical Algorithm
CPG	Clinical Practice Guideline
DBP	Diastolic Blood Pressure
DM	Diabetes Mellitus
DT	Decision Tree
ECG	Electrocardiography
EOC	Episode Of Care
GOT	Glutamyl Oxaloacetic Transaminase
GPT	Glutamyl Pyruvic Transaminase
HT	Hypertension
LPO	Layered Partial Order
LVH	Left Ventricular Hypertrophy
OHD	Oral Hypoglycemic Drug
SBP	Systolic Blood Pressure
SDA	State Decision Action

1

Introduction

Clinical Practice Guidelines (CPGs) (BGK$^+$02) are medical documents that guide decisions and criteria regarding diagnosis, management, and treatment in specific areas of health-care based on an examination of current evidence. In general, they gather all the available evidence related to a disease. Their main aim is to support and promote good clinical practice but they are also used to provide a homogeneous practice, to improve the quality, the equality and the equity of patient care, to avoid several kinds of risk and to reduce costs. Most of the CPGs include Clinical Algorithms (CAs) (SfMDM92; Had95) which are flowcharts that graphically summarize some of the medical procedures described in the guideline.

CPGs are usually produced at national or international levels by medical associations or governmental bodies which are based on scientific rigor, employing systematic reviews and meta-analyses (GR93). Generally, the development of CPGs represents a laborious task that implies the cooperation of several health care experts of different specialties. Moreover, the manual generation of CPGs suffers from other drawbacks due to the individual differences of each patient or the differences on the treatment depending on the presence of comorbidities (VSS$^+$09).

Some approaches have been carried out to *automatically induce* part of the knowledge contained in CPGs from the hospital databases, specially *procedural knowledge* (MSL$^+$08). These approaches are not necessarily based on the medical evidence but on the experience of the medical daily practice. Some work has been done on the induction of clinical pathways represented as Petri nets (MSSvdA08; MSL$^+$08) which is based on a technique called workflow mining (vdAvDH$^+$03). However, the structures induced by

those systems are not explicit medical structures that doctors are familiar with. Other approaches (RLVT08; BRLV12) directly generate CAs which are pure medical structures. Although these approaches have obtained good results in concrete medical applications, they suffer from two major drawbacks. On the one hand, the current algorithms to induce CAs are only based on statistical measures and do not consider any kind of medical or clinical *background knowledge* which is not explicit in the hospital databases. However, attending to the indications of this knowledge it is essential to guarantee medical correct and comprehensible structures (LVRC07; LVRC12b; LVRB12). On the other hand, they are not able to deal with *incremental* data which is of great importance in medicine because the hospital databases are being constantly updated with new information generated in daily clinical practice.

The objective of this thesis is to propose a methodology to induce medically correct and comprehensible CAs represented using the *State Decision Action (SDA) knowledge model* (Ria07; BRLV12) from hospital databases that fulfill the *Episode Of Care (EOC) data model* (Ria10). The methodology considers the relevant *background knowledge* of the domain, previously validated by health care experts, which is represented using different kinds of structures: graphs, hypergraphs, layered partial orders and concept hierarchies. Moreover, the methodology is able to work in an *incremental* way, so that the CAs generated are updated as soon as new data arrives.

The main *contributions* of the thesis are the following:

Contribution 1: Formalization of background knowledge structures that support the automatic induction of medically correct and comprehensible CAs.

Contribution 2: Construction of a background knowledge repository for the diseases of hypertension (ohhs03), diabetes mellitus (CL08) and the comorbidity of both diseases (MSRS07).

Contribution 3: Development of both an incremental and a non incremental methodology to induce medically correct and comprehensible CAs based on medical background knowledge.

Contribution 4: Application of these methodologies to automatically generate SDA diagrams representing correct and comprehensible CAs for the long term management of hypertension, diabetes mellitus and the comorbidity of both diseases in the primary care centers of the SAGESSA Health-Care Group (SAG).

The different hospital databases of patients and other medical resources used in this thesis belong to the SAGESSA Health-Care Group. During the development of this thesis we have been assisted by health care professionals from SAGESSA who were one of the main sources of expertise and experience during the knowledge acquisition process, previous to the construction of the repository of background knowledge. These health care professionals also supervised the medical validity of the thesis and they helped in the analysis of the partial and the final results obtained.

The structure of this document is the following:

Chapter 1: An introduction to the thesis, its general objective, the main contributions and the structure of the document.

Chapter 2: Analysis of the current state of the art of the different techniques, methodologies, structures and models related to the topics of this thesis.

Chapter 3: Formalization of the medical background knowledge needed to solve each one of the problems involved in the automatic induction of CAs (contribution 1) and presentation of a repository of background knowledge for hypertension, diabetes mellitus and the comorbidity of both diseases (contribution 2).

Chapter 4: Proposal of non-incremental and incremental algorithms to solve the two main problems in the automatic induction of CAs using background knowledge, and their integration into a global methodology to generate CAs (contribution 3).

Chapter 5: Set of tests regarding the performance of the background knowledge and incrementality, the adherence of the results to the database and related medical issues, the analysis of the results and the presentation of the final SDA diagrams for three chronic pathologies (contribution 4).

Chapter 6: Conclusions and final discussion about the thesis.

Chapter 7: Future research lines.

2

State of the art

The automatic *induction of procedural knowledge* from hospital databases is a medical informatics problem which has been object of research using several approaches in the bibliography. In our approach, the knowledge structures used are *SDA diagrams* representing clinical algorithms about long term treatments. These SDA diagrams are induced from hospital databases that follow the *EOC data model*. The induction of these knowledge structures follows a complex procedure involving *clustering* and *decision making* techniques. During the induction of SDA diagrams, we consider *background knowledge* of the domain in order to assure the medical correctness of the results. This background knowledge may be represented using tools like *graphs*, *partial orders*, *concept hierarchies*, etc. Moreover, if the induction of SDA diagrams is made *incremental*, our methods will be able to efficiently cope with the arrival of new data to the hospital databases.

All the concepts and techniques involved in the knowledge-based, incremental, and automatic induction of SDA diagrams are presented in the following sections together with a review of the most relevant related bibliography.

2.1 Clustering

Clustering is the mathematical problem of, given a set of patterns, classify them into classes (clusters), so that similar patterns belong to the same cluster and dissimilar ones belong to different clusters.

Formally speaking, a clustering problem $(\mathcal{X}, C)$ consists of a domain of patterns $\mathcal{X}$ and a set of clusters C. Each cluster $C_i \in C$ contains a subset of the patterns in $\mathcal{X}$.

Clustering is a common practice in health care, for example classifying profiles of patient, defining health care patient states within the treatment of a disease, grouping different kinds of features about patients, risk factors or drugs.

2.2 Decision making

Decision making is the mathematical problem of, given a situation, choosing one action out of a set of actions to be performed for that situation.

Formally speaking, a decision problem $(\mathcal{X}, D, f)$ consists of a domain $\mathcal{X}$, a set of decisions D, and a decision function $f : \mathcal{X} \to D$. The problem of decision is equivalent to the problem of classification (Jam86).

In medicine, decisions are made continuously with different purposes: screening, diagnosing, prognosing, drug and therapy prescription, etc. Through the years, multiple structures have been proposed to formalize these decision processes. They range from statistical approaches as Bayesian Networks (LvdGAH04; VdCFL07) or probabilistic models (DH04) to symbolic approaches as decision trees (GC03; PKSR02; Tur09), decision tables (Shi97) or decision rules (CN89).

2.3 Clustering algorithms

The clustering problem has been widely treated in artificial intelligence. It is raised as the unsupervised classification of instances (patterns, observations, data items, or feature vectors) into classes (clusters) usually based on a proximity measure or, in a more general way, on the properties that data share.

Formally, an instance x is a single data item which consists of a vector of m measurements $x = (a_1(x), ..., a_m(x))$. Each position i of the vector corresponds to a certain attribute or feature a_i which has a value $a_i(x)$ for the instance x. An instance set (or data set) is denoted $\mathcal{X} = \{x_1, ..., x_n\}$ where the ith instance is denoted $x_i = (a_1(x_i), ..., a_m(x_i))$. Each instance x_i is classified into a cluster C_j of a finite set of clusters $C = \{C_1, ..., C_k\}$. A cluster can be viewed as a source of instances whose

distribution in the space of features is governed by a probability density specific to the cluster.

Clustering is used to solve very different kinds of problems, thus the bibliography is full of different clustering algorithms. Several attempts have been made to classify clustering algorithms (JMF99; KP04; XW05). One of the most typical classifications is between hard-clustering (McQ67; KR90; Hua98), fuzzy-clustering (Dun73) or overlapping-clustering (Lel94; RD00; Cle04; BKG$^+$05; CH06; FGK$^+$09).

In *hard-clustering* each instance is assigned to a single cluster. We attempt to seek a k-partition of $\mathcal{X}$, $C = \{C_1, ..., C_k\}$ $(k \leq n)$ such that:

- $C_i \neq \emptyset$, $i = 1, ..., k$
- $\bigcup_{i=1}^{k} C_i = \mathcal{X}$
- $C_i \cap C_j = \emptyset$, $i, j = 1, ..., k$ and $i \neq j$

Some examples of hard-clustering can be found in (McQ67; KR90; Hua98).

Conversely, the *fuzzy-clustering* methods propose an organization in which each instance participates to the definition of each cluster. An instance is allowed to belong to all clusters with a degree of membership $u_{i,j} \in 0..1$, which represents the membership coefficient of the jth instance in the ith cluster and satisfies $\sum_{i=1}^{k} u_{i,j} = 1, \forall j$ and $\sum_{j=1}^{n} u_{i,j} < n, \forall i$. The most popular fuzzy clustering algorithm is fuzzy c-means (FCM) (Dun73).

Finally, in *overlapping-clustering* rather than assigning an instance to only one cluster, we allow an instance to belong to one or several clusters, final clusters thus intersect. We attempt to seek a covering of $\mathcal{X}$, $C = \{C_1, ..., C_k\}$ $(k \leq n)$ such that:

- $C_i \neq \emptyset$, $i = 1, ..., k$
- $\bigcup_{i=1}^{k} C_i = \mathcal{X}$
- $x \in C_i \not\Rightarrow x \notin C_j$, $i, j = 1, ..., k$ and $x \in \mathcal{X}$

Some examples of overlapping-clustering are axial-k-means algorithm (Lel94), pyramidal clustering (RD00), PoBOC (Cle04), MOC (BKG$^+$05), OPC (CH06) and the graph-based clustering method in (FGK$^+$09).

Another typical classification of clustering algorithms is among partitioning methods and hierarchical methods, although several less important approaches have been carried out like density-based methods, grid-based methods or model-based methods.

Partitioning methods classify n instances into k clusters, where k is specified by the user. The partitional techniques usually produce clusters by optimizing a criterion function defined either locally (on a subset of the instances) or globally (defined over all of the instances). The most intuitive criterion function in partitional clustering techniques is the squared error criterion. The k-means (McQ67) is the simplest and most commonly used algorithm employing a squared error criterion. It starts with a random initial partition and keeps reassigning the instances to clusters based on the similarity between the instance and the cluster centers until a convergence criterion is met. Other similar methods are k-medoids (or PAM), CLARA (Clustering LARge Applications) (KR90) and k-modes (Hua98).

Hierarchical clustering builds a cluster hierarchy also known as a dendrogram. Every cluster node contains sibling clusters which partition the points covered by their common parent. Such an approach allows exploring data on different levels of granularity. Hierarchical clustering methods are categorized into agglomerative (bottom-up) as AGNES (AGglomerative NESting) (KR90) and divisive (top-down) as DIANA, Divisive Analysis (KR90). An agglomerative clustering starts with one-point (singleton) clusters and recursively merges the two (or more) most appropriate clusters. A divisive clustering starts with one cluster of all data points and recursively splits the most appropriate cluster. The process continues until a stopping criterion (frequently, the requested number k of clusters) is achieved. Some examples of other kinds of hierarchical clustering algorithms are COBWEB (Fis87), CURE (SGS98), BIRCH (TZL96), Chameleon (GKK99), ROCK (GRS00) and the overlapping pyramidal clustering (RD00).

Clustering is a common technique applied in lots of domains in biomedicine. It has been used in bioinformatics in order to group genes (TBK09), in medical image processing (MS99) but also in the health care domain. Some work has been done applying clustering methods to group different kinds of risk factors (SvLTO02; RGO$^+$04; Poo07; AP09) and respiratory parameters (BAL$^+$01). Concerning diseases, in (MAEB04) spacetime clustering is examined amongst cases of lymphoma in children. (KN08) proposes a new clustering technique based on Ant Colony Optimization which is used to classify types of arrhythmia. A conceptual clustering method is applied over cancer

data in (dV96). And (PAKS09) presents Onto-clust, a methodology that combines clustering analysis and ontological methods in order to identify groups of comorbidities for developmental disorders.

2.4 Decision making algorithms

Artificial intelligence has a long tradition inducing decision functions. Having a set of instances (patterns, observations, data items, or feature vectors) and a set of decisions (classes, or actions), a decision function is automatically learned such that given a certain instance it is able to propose a decision. This methodology is also referred to as supervised classification.

Formally, an instance x is a single data item which consists of a vector of m questions and one decision $x = (q_1(x), ..., q_m(x), d_x)$. Each position i of the first m positions of the vector corresponds to a certain question or test q_i which has a value $q_i(x)$ for the instance x. The last position corresponds to a certain decision d_x for this instance. An instance set (or data set) is denoted $\mathcal{X} = \{x_1, ..., x_n\}$ where the ith instance is denoted $x_i = (q_1(x_i), ..., q_m(x_i), d_{x_i})$.

There are several mechanisms to represent decisions, which can be induced from data (KZP06). Some mechanisms are logic-based as decision trees (PKSR02; GC03; Tur09) and decision rules (CN89), some are perceptron-based as single (FS99) or multi layered (Zha00) perceptrons and RBF networks (RH01), some are statistical approaches as Bayesian Networks (LvdGAH04; VdCFL07) and some others are instance-based (CH67) or support vector machines (Bur98). In this thesis we focus on Decision Trees (DTs).

A tree is a mathematical concept that denotes a simple, undirected, connected and acyclic graph. The edges are known as branches, the vertices of order 1 are called leaves and the rest of the vertices, internal nodes. A rooted tree is a tree in which a special node is singled out. This node is called root. In such kind of trees, nodes which are one edge away from a given node n are called successors of n.

DTs (PKSR02; GC03; Tur09) are rooted trees used as decisional structures to solve a decisional problem $(\mathcal{X}, D, f)$. Each internal node contains a question q_i taken from a set of questions Q on the elements of $\mathcal{X}$ that represents the function $q_i : \mathcal{X} \to \Delta_i$ such that any element $x \in \mathcal{X}$ is given an answer $q_i(x)$ in the answer domain Δ_i. Each

internal node with question q_i represents a partition of the domain Δ_i and it has as many successors as parts are in that partition. Each branch leading from an internal node q_i to a successor of q_i is labeled with one of the possible parts of the partition that the internal node represents. The leaves of the DT contain final single decisions from the set D.

Figure 2.1 depicts an example of a simple DT for the treatment of hypertension. It contains sequences of questions that lead to two alternative decisions *Lifestyle modification* and *Lifestyle modification+Pharmacological treatment*. The DT uses question *stage*, meaning the stage of Hypertension (HT) in which the analyzed patient is, with answers *pre-HT, 1* and *2, 3* and question *risk group*, meaning the level of health fragility of the analyzed patient, with answers *A, B* and *C*.

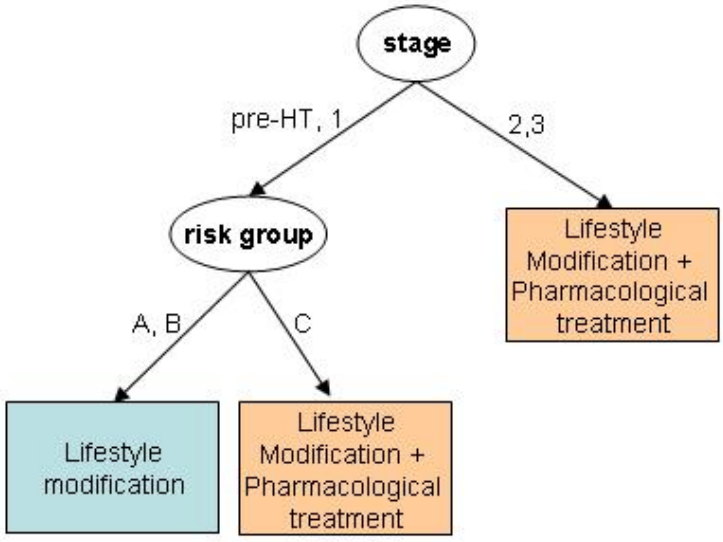


Figure 2.1: Example of decision tree

In a decision problem $(\mathcal{X}, D, f)$, a DT may act as the decision function f. Given an instance $x \in \mathcal{X}$, such that $q_i(x) = v_i$ where v_i is the answer to question q_1 for x, the DT determines a path from the root node to a certain leaf. To decide which path corresponds to instance x, at each internal node n_i containing a question q_i, the next node of the path $n_{i'}$ is the successor connected to n_i by the branch labeled δ_j such that $\delta_j \in \Delta_i$ and $v_i \in \delta_j$ (i.e., at each internal node the branch whose label matches the answer of the question for x is followed). This process is repeated until the path is

completed and, thus, a leaf with a final decision d is reached.

Each path from the root to a leaf might be seen as a constraint over the answers of the questions in the path expressed as a conjunction. It is equivalent to a rule $\{q_1(x) \in \delta_1\} \wedge \{q_2(x) \in \delta_2\} \wedge \ldots \wedge \{q_n(x) \in \delta_n\} \rightarrow d_x$ where q_i is the question corresponding to the ith node of the path, each δ_i is one of the different parts of the partition Δ_i, and $d_x \in D$ is the decision contained in the leaf confirming that all the elements $x \in \mathcal{X}$ arriving to this leaf satisfy $f(x) = d_x$.

For example, the DT in figure 2.1 is used to make a decision on a new instance (patient) x starts asking question *stage*. If *stage(x)=1* the instance goes down through the left branch labeled *pre-HT, 1.* According to the DT, the next question to ask is *risk group*. Supposing that *risk group(x)=A* we finally make decision *Lifestyle modification.*

Although DTs can be build by hand, one of the main processes to automate the construction of DTs is by induction or supervised learning. This means that, given a set of instances $\mathcal{X}$ and a set of decisions D, the DT is automatically built using an inductive learning algorithm.

The various heuristic methods to induce DTs can be divided into top-down approaches and bottom-up approaches, although other alternatives exists like hybrid and tree growing-pruning approaches (SL91).

Top-down is the strategy of starting from the root node and generating the successive internal nodes of the tree until reaching the leaves. All the data set used to generate the DT is placed in the root node and then one of the questions is selected to split them into the different branches of the node. This selection is done by means of a node splitting rule. The procedure is repeated for the successors and so on until the algorithm decides to place a leaf. For example, when a certain percentage of the instances have the same decision. When a leaf is placed, a decision is made for all the instances in this leaf. Usually, the decision made is the one with the highest probability. Most of the research is concentrated in the area of finding the splitting rule. One of the main approaches followed is using the information gain (Qui86; Qui93). The information gain criterion measures the amount of information (SW48) gained by partitioning the data set according to the answers of a single question. This is to say, the information gain of a given question q_i with respect to the decision set D is the reduction of uncertainty about the decision to make when the value of q_i is known. The

uncertainty about the decision set D is measured by the *entropy* $E(D)$. We define the entropy of D in equation 2.1.

$$E(D) = -\sum_{i=1}^{m} Pr(D = d_i) \log_2(Pr(D = d_i)) \tag{2.1}$$

If the value of q_i is already known, we define the uncertainty about D by the conditional entropy of D given q_i, $E(D|q_i)$ in equation 2.2.

$$E(D|q_i) = \sum_{\delta_j in \Delta_i} Pr(q_i = \delta_j) E(D|q_i = \delta_j)) \tag{2.2}$$

Thus, the information gain of q_i with respect to D is defined as equation 2.3.

$$I(D; q_i) = E(D) - E(D|q_i) \tag{2.3}$$

One of the most famous DT inductive algorithms based on the concept of information gain is ID3 (Qui86). Essentially, it builds the tree by computing at each internal node the information gained when splitting the data set using each of the questions and selecting the one that maximizes the gain. The main disadvantage of using the information gain is that it has a strong bias in favor of the questions with many answers. Another algorithm called C4.5 (Qui93) solves this by using the so-called gain ratio criterion. The idea is to use the gain ratio $GR(D; q_i)$ in equation 2.4 to select the best question instead of the information gain.

$$GR(D; q_i) = \frac{I(D; q_i)}{S(D; q_i)} \tag{2.4}$$

In equation 2.4, $S(D; q_i)$ is known as the split information which is sensitive to how wide and uniform the partition induced by a question is. It is defined in equation 2.5.

$$S(D; q_i) = \sum_{\delta_j in \Delta_i} Pr(q_i = \delta_j|D) \log_2 \frac{1}{Pr(q_i = \delta_j|D)} \tag{2.5}$$

The C4.5 algorithm has another important improvement with respect to ID3. This is the incorporation of pruning strategies to simplify the DT and to avoid overfitting.

In *bottom-up approaches* (LTG$^+$83) the DT is constructed using some distance measure, such as Mahalanobis-distance or pair-wise distances between a priori defined classes. Then, in each step the classes with the smaller distance are merged to form a

new group. The mean vector and the covariance matrix for each group are computed from the training samples of classes in that group, and the process is repeated until one is left with one group at the root. This has some of the characteristics of a hierarchical clustering approach.

In health care, the problem of decision is so frequent that sometimes it receives special names as diagnosing (i.e., decide on the sort of disease), assessing a patient condition (i.e., decide the severity of a disease sign or symptom) or prescribing a treatment (i.e., decide on the proper therapy). In the case of DTs, they have been widely applied to solve medical problems (PKSR02; GC03). For example, DTs have been used to predict cesarean delivery (SMC$^+$00), to classify patients as having acute cardiac ischemia (LGSD93) or to enhance the tuberculosis prevention in nontraditional settings and relationships (KMB$^+$05).

2.5 Background knowledge

In machine learning, *background knowledge* (or prior knowledge) is any information that can be fed by an expert of the domain or that can be extracted from a source of knowledge, which is not explicitly included in the input data source of the learning process. Background knowledge is used in complex domains, like medicine, where the input data source is not enough informative to obtain correct results. This background knowledge may contain constraints that must be fulfilled, relations or orderings between variables, additional criteria that must be taken in account, labeled data, expert's settings of learning parameters, etc. Including background knowledge in the design of learning methods is important. Firstly, the role of the expert in asserting background knowledge imposes certain requirements to the way knowledge can be expressed. Attention has to be paid to aspects of understandability for expressing the required background knowledge. On the other hand, the use of prior knowledge also creates some new problems. There is the question of priority. In case of conflict or contradiction between a partial solution being built from data and the knowledge expressed at the beginning of the learning process, background knowledge can help to resolve these conflicts and also to ensure that prior knowledge does not preclude the extraction of really useful models (SC00).

In section 2.5.1 we introduce the tools that have been used in this thesis to represent background knowledge, then in section 2.5.2 and section 2.5.3 we make a brief survey on how background knowledge has been incorporated in clustering and decision making, respectively.

2.5.1 Representation of background knowledge

The way that background knowledge is represented must be simple so that it can be easily understood by the expert, but it also must be complex enough to gather all the required knowledge. In the bibliography, background knowledge has been represented by different means such as graphs, partial orders, concept hierarchies, cost functions, etc. (LYWZ04; LWY04; COR05; SAM05; BJ06; SY06; MCKD$^+$06; FCPB07; LVRC07; VMFC09; LVRC12b; LVRB12; XW12).

2.5.1.1 Graphs and hypergraphs

In mathematics, a *graph* refers to a collection of points called *vertices* (or nodes) and a collection of lines called *edges* connecting pairs of vertices. Any binary relation is a graph, so graphs can be used to represent essentially any relationship.

Definition 2.5.1 (Graph) *A graph G is defined as an ordered pair $G = (V, E)$ comprising a set V of vertices and a set E of edges. An edge is defined as a pair of vertices $v_i, v_j \in V$.*

In an *undirected graph* the edges are unordered pairs $\{v_i, v_j\}$, so the relations between pairs of vertices are symmetric and they have no directional character. In a *directed graph* (or digraph) the edges are ordered pairs (v_i, v_j) having a directional character and which are represented as arrows. A *labeled graph* is a graph such that its edges have a label that gives them a different meaning. Figure 2.2 shows an undirected graph where $V = \{IN,BN,SBDC,SBD,SBDF,SCDF,SCD,SCDC\}$ and $E = \{\{IN,BN\},\{BN,SBDC\},\{SBDC,SBD\},\{SBD,SBDF\},\{SBDF,SCDF\}, \{SCDF, SCD\}, \{SCD,SCDC\},\{SCDC,IN\},\{IN,SCDF\},\{BN,SBDF\},\{SCDC,SBDC\},\{SCD,SBD\},\{SCDC,SBD\},\{SCDC,SBDC\}\}$. This graph is a representation of the direct relations between pharmacological entities of the RxNorm, extracted from (BP09).

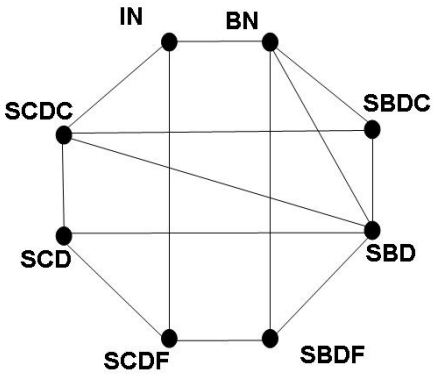


Figure 2.2: Example of undirected graph

Figure 2.3 depicts a graphical representation of a directed labeled graph where $V = \{Patient, AngioVisit, CoronaroExam, Symptom, RiskFactor\}$ and $E = \{(Patient, AngioVisit), (AngioVisit, CoronaroExam), (Patient, Symptom), (Patient, RiskFactor)\}$. This graph is a subpart of the Multimedia Temporal Graphical Model presented in (COR05).

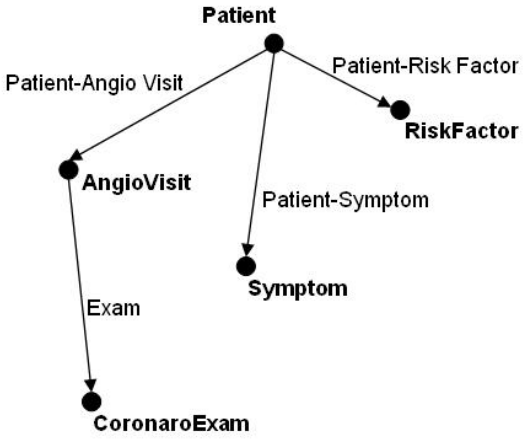


Figure 2.3: Example of directed labeled graph

There is an alternative tabular representation which is useful when dealing with large graphs. For example, table 2.1 contains the graph in figure 2.2. For each pair of vertices we mark with 'X' if they are related. In this case, as this is an undirected graph, we only need to fill the cells above the main diagonal (the cells below the main diagonal represent the same relationships). On the contrary, for directed graphs, if we have the edge (v_i, v_j) we would mark the cell in the row of v_i and the column of v_j,

while if we have (v_j, v_i) we would mark the cell in the row of v_j and the column of v_i. With labeled graphs, we mark the cells with the label of the edge rather than with 'X'.

Table 2.1: Example of undirected graph represented as a table

	IN	BN	SBDC	SBD	SBDF	SCDF	SCD	SCDC
IN		X				X		X
BN			X	X	X			
SBDC				X				X
SBD					X		X	X
SBDF						X		
SCDF							X	
SCD								X
SCDC								

A *hypergraph* is a generalization of a graph, where an edge can connect any number of vertices.

Definition 2.5.2 (Hypergraph) *A hypergraph H is defined as an ordered pair $H = (V, E)$ comprising a set V of vertices and a set E of hyperedges. A hyperedge is defined as a non-empty subset of V. Therefore, E is a subset of $\mathcal{P}(V) - \emptyset$.*

Figure 2.4 depicts two different graphical representations of a same hypergraph where $V = \{EGF, BRCA1, BRCA2, RB, RAS, myc\}$ and $E = \{\{EGF, BRCA1, BRCA2, RAS\}, \{BRCA1, BRCA2, RB\}, \{RAS, myc\}\}$. This is a subpart of a hypergraph presented in (MPM10) representing breast cancer knowledge. In the hypergraph, genes $BRCA1$, $BRCA2$ and RB are connected with a relationship called *tumor suppressor gene*, EGF, $BRCA1$, $BRCA2$ and RAS with a *breast cancer* relationship, and myc and RAS with a *oncogene* relationship.

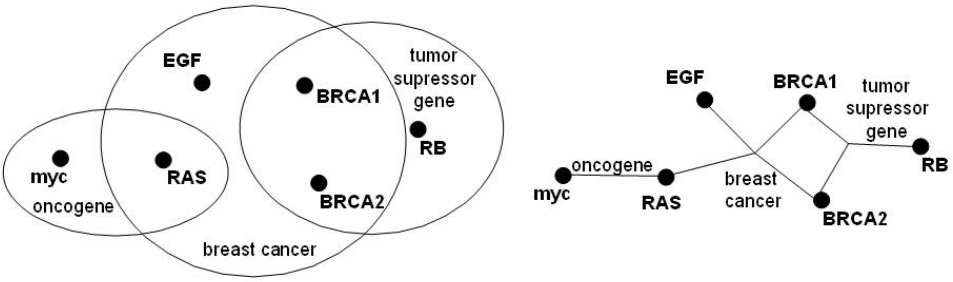


Figure 2.4: Example of labeled hypergraph

If we are dealing with large hypergraphs we can represent them in a tabular way (see table 2.2). Column 1 contains the name of the hyperedge and column 2, the related vertices. The horizontal lines separate the different sets of related vertices.

Table 2.2: Example of labeled hypergraph represented as a table

breast cancer	EGF
	BRCA1
	BRCA2
	RAS
oncogene	myc
	RAS
tumor suppressor gene	RB
	BRCA1
	BRCA2

Several concepts of graph theory have been widely applied to solve several kinds of problems in biomedicine (BCM+05; RWL+08; BP09; LXH+10; MPM10). Concretely in health care, (BCM+05) has applied graph theory to identify instances of deprivation and high morbidity and mortality in health data sets and in (LXH+10) to study adverse drug events. Graphs (in some cases simply referred to as relationships between elements) are also becoming a common tool to represent biomedical knowledge (COR05; BJ06; VMFC09; XW12). In (COR05) graphs are used in the representation of a clinical database for cardiology patients. In (XW12), background knowledge, represented using drug-gene relationships, is used to improve the extraction of pharmacogenomics specific drug gene relationships from free text.

2.5.1.2 Partial orders and LPOs

In mathematics, a partially ordered set (or poset) formalizes the intuitive concept of an ordering, sequencing, or arrangement of the elements of a set.

Definition 2.5.3 (Partial order) *A relation* $\leq$ *is a partial order on a set* S *if it has:*

1. *Reflexivity:* $a \leq a$ *for all* $a \in S$
2. *Antisymmetry:* $a \leq b$ *and* $b \leq a$ *implies* $a = b$, $a, b \in S$
3. *Transitivity:* $a \leq b$ *and* $b \leq c$ *implies* $a \leq c$, $a, b, c \in S$

Definition 2.5.4 (Partially ordered set) *A partially ordered set P is defined as an ordered pair $(S, \leq)$ where S is called the* ground set *of P and $\leq$ is a partial order on the set S.*

Two elements $a, b \in S$ are *comparable* if either $a \leq b$ or $b \leq a$ or both.

Given three elements $a, b, c \in S$ such that $a \leq b \leq c$ then b is said to be *between* a and c.

Definition 2.5.5 (Cover) *Given a partially ordered set $(S, \leq)$ and two elements $a, b \in S$ we say a covers b if $a \leq b$ and there is not any $c \in S$ such that $a \leq c \leq b$ or if $b \leq a$ and there is not other $c \in S$ such that $b \leq c \leq a$. In the first case, a is a lower cover of b ($a \prec b$) and in the second case a is an upper cover of b ($b \prec a$). We denote $c(a)$ the set of covers of a (i.e., $c(a) = \{b \in S : a \prec b \text{ or } b \prec a\}$).*

Definition 2.5.6 (Chain) *A subset $C \subseteq S$ is called a chain in $P = (S, \leq)$ if and only if for any pair $a, b \in C$, $a \leq b$ or $b \leq a$ or both (i.e., C is a totally ordered subset of S).*

Definition 2.5.7 (Length) *The length of a partially ordered set $P = (S, \leq)$ is the cardinality of the biggest chain in P.*

Definition 2.5.8 (Antichain) *A subset $C \subseteq S$ is called antichain in $P = (S, \leq)$ if and only if for any pair $a, b \in C, (a \neq b)$ neither $a \leq b$ nor $b \leq a$ (i.e., C is a totally unordered subset of S).*

Definition 2.5.9 (Width) *The width of a partially ordered set $P = (S, \leq)$ is the cardinality of the biggest antichain in P.*

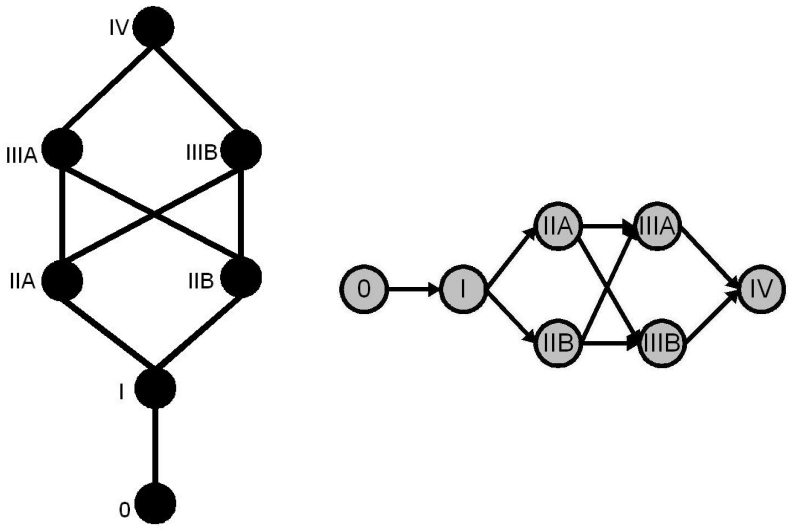
Definition 2.5.10 (Maximal (minimal) element) *Given a partially ordered set $(S, \leq)$, an element $a \in S$ is called a maximal (or minimal) element if there is none $b \in S$ for which $a \leq b$ (or $b \leq a$). The set of maximal (or minimal) elements of a partially ordered set $P = (S, \leq)$ is denoted $MAX(P)$ (or $MIN(P)$).*

Hasse diagrams (PS03) are used to represent partially ordered sets. These diagrams are a graphical rendering of a partially ordered set displayed via the cover relation of the partially ordered set with an implied upward orientation. A point is drawn for each element of the ground set of the partially ordered set, and line segments are drawn between these points according to the following two rules:

1. If $a \leq b$ in the partially ordered set, then the point corresponding to a appears lower in the drawing than the point corresponding to b.

2. The line segment between the points corresponding to any two elements a and b of the partially ordered set is included in the drawing if and only if $b \in c(a)$.

Figure 2.5 depicts an example of a Hasse diagram of the partially ordered set with $S = \{0, I, IIA, IIB, IIIA, IIIB, IV\}$ representing stages of breast cancer according to (BRR07). It is observed, for example, that $I \leq IIA$ because I appears in a lower position of the diagram and there is a way of going up from I to IIA. This represents the meaning that stage I is better than stage IIA, as far as breast cancer is concerned. By transitive property, we may conclude also that $I \leq IIIB$, $0 \leq IV$, etc. but it is impossible to say whether $IIA \leq IIB$ or $IIB \leq IIA$.



(a) Hasse diagram (b) Left-right Hasse diagram

Figure 2.5: Example of partially ordered set

In health care, these diagrams are usually represented as left-right Hasse diagrams as the one depicted in figure 2.5(b) obtained from (BRR07).

Given a partially ordered set $P = (S, \leq)$ the order relation $\leq$ is called a *preorder* or a *quasiorder* if it does not necessarily satisfy the antisymmetry property.

Definition 2.5.11 (Preorder) *A relation $\leq$ is a preorder on a set S if it has:*

1. *Reflexivity:* $a \leq a$ *for all* $a \in S$
2. *Transitivity:* $a \leq b$ *and* $b \leq c$ *implies* $a \leq c$, $a, b, c \in S$

Notice that partial orders are particular cases of preorders.

Given a partially ordered set $P = (S, \leq)$ the order relation $\leq$ is called a *total order* if all the elements are comparable with $\leq$.

Definition 2.5.12 (Total order) *A relation* $\leq$ *is a total order on a set* S *if it has:*

1. *Reflexivity:* $a \leq a$ *for all* $a \in S$
2. *Antisymmetry:* $a \leq b$ *and* $b \leq a$ *implies* $a = b$, $a, b \in S$
3. *Transitivity:* $a \leq b$ *and* $b \leq c$ *implies* $a \leq c$, $a, b, c \in S$
4. *Comparability (trichotomy law):* $\forall a, b \in S, \{a \leq b \vee b \leq a\}$

A set S equipped with a total order relation $\leq$ is called a *totally ordered set* or *linearly ordered set*. Notice that total orders are particular cases of partial orders.

In medicine, an example of partial order is the previously mentioned stages of breast cancer (BRR07), an example of preorder is the preference of selection of questions during an encounter according to the health risk of the medical test needed to answer them and an example of total order is fever. In the first case, the possible stages of breast cancer are 0, I, IIA, IIB, IIIA, IIIB and IV and the comparison of stages among the patients must be made according to figure 2.5 and it is not always possible to determine which patient is worst according to these stages (e.g., patients in stage IIA are not comparable with patients in IIB). In the second case, a question q_1 entails more risks than a question q_2 (i.e., $q_2 \leq q_1$) or their risks can be uncertain (i.e., neither $q_1 \leq q_2$ nor $q_2 \leq q_1$). But if q_1 and q_2 are obtained using a same medical test we know that the health risk of q_1 and q_2 is exactly the same (i.e., $q_1 \leq q_2$ and $q_2 \leq q_1$). In the third case, fever can be expressed as the body temperature in Celsius degrees ($^\circ$ C) or by the terms normal (36.5-37.5 $^\circ$ C), hypothermia (<35.0 $^\circ$ C), fever (37.5-38.3 $^\circ$ C), hyperthermia (37.5-38.3 $^\circ$ C) and hyperpyrexia (40.0-41.5 $^\circ$ C) and, therefore, it is always possible to determine which is the patient with a higher temperature among a group of patients.

Partial orders have many applications in biomedicine. Partial orders can be induced from medical data in order to extract knowledge about implicit precedences among

these data. For example, (BRR07) describes an algorithm that uses data about patient-professional encounters in order to induce partial orders on the patient conditions of a disease. In (SA04), partial orders are induced from an ontology of biomedical terms and in (UFM05) partial orders are extracted from unordered 0-1 data in the domain of medical genetics. Moreover, partial orders can be used to represent background knowledge in medical decision making. In (LVRC07), a partial order representing the medical protocol of asking questions in the treatment of a certain disease is used to increase the acceptability of decision trees in medicine and in (LVRB12) the same problem is solved by means of partial orders which may represent several health care criteria such as economic cost, health-risk or comfortability.

Definition 2.5.13 (Layered partial order) *A Layered Partial Order (LPO) is a partial order $\leq$ that satisfies the following property:*

- $\forall a, b \in S$, *if* a *and* b *are not related (i.e., neither* $a \leq b$ *nor* $b \leq a$*), then* $c(a) = c(b)$

They are called layered partial orders because the elements in S are strictly arranged in layers. A layered partial order determines n disjoint antichains: $C_1, C_2, ..., C_n$ such that $\bigcup_{i=1}^{n} C_i = S$ and for each pair of elements $(a, b) \in C_i \times C_j$, where $i < j$, we have that $a \leq b$. Notice that if $j = i + 1$, we also have that $a \prec b$. Each antichain C_i $(i = 1..n)$ is called a *layer* of the partial order. An element $a \in S$ is in layer i-th if $a \in C_i$. The layer of an element $a \in S$ is denoted as $\ell(a)$ and it is represented by a positive number. A layered partial order with n layers is called n-layered partial order (n-LPO).

For example, the partial order depicted in figure 2.5 is an LPO (concretely a 5-LPO). As it can be observed, the elements of this partial order are arranged in the following layers:

$$\{0\}, \{I\}, \{IIA, IIB\}, \{IIIA, IIIB\}, \{IV\}$$

Observe that the covers of the elements in the same layer are identical and different from the covers of the elements in other layers. For example, $c(IIA) = c(IIB) = \{I, IIIA, IIIB\}$ and $c(IIIA) = \{IIA, IIB, IV\}$. The elements within each layer are not related to each other (e.g., $IIA \not\leq IIB$ and $IIB \not\leq IIA$) and they are all related to each of the elements of the next (or previous) layers (e.g., IIA is related to each

element of the previous layers ($0 \leq IIA$, $I \leq IIA$) and to each element of the next layers ($IIA \leq IIIA$, $IIA \leq IIIB$, $IIA \leq IV$).

Due to its structuring in strict layers of priority, LPOs can be easily represented as tables. For example, table 2.3 represents the same LPO of figure 2.5. Column 1 contains the number of the layer (ℓ) and column 2, the elements in each layer. The horizontal lines separate the different layers of the LPO.

Table 2.3: Example of LPO represented as a table

1	0
2	I
3	IIA
	IIB
4	IIIA
	IIIB
5	IV

It is common to use LPOs to represent medical knowledge that involves partial orders (BRR07; LVRC07; LVRB12). Their composition in layers of priority makes them easier to understand and also more natural to medical problems. Therefore, in the rest of the document, when we are referring to a partial order we are actually meaning an LPO.

2.5.1.3 Concept hierarchies

Concept hierarchies (taxonomies or is-a hierarchies) organize data or concepts in hierarchical forms expressing knowledge in concise, high level terms, and facilitating mining knowledge at multiple levels of abstraction (Lu97). Although they are defined as partially ordered sets, they are explained in an independent section as they usually do not represent an ordering but semantic relationships of subsumption.

Definition 2.5.14 (Concept hierarchy) *A concept hierarchy $\mathcal{H}$ is a partially ordered set $(H, \leq)$ where H is a finite set of concepts and $\leq$ represents a subsumption relationship between concepts.*

The partial order $\leq$ of a concept hierarchy reflects the special general relationship between concepts, which is also called subsumption relation, subconcept-concept relation or *is-a relation*. If $a \leq b$ we say that a is a subconcept of b and b is a superconcept of a.

Since partially ordered sets can be visually sketched using Hasse diagrams, we can also use this kind of diagrams to express concept hierarchies. In the case of concept hierarchies, their Hasse diagrams are usually drawn upwards so if $a \leq b$ then the point corresponding to a appears higher in the drawing than the point corresponding to b. Figure 2.6 depicts an example of a concept hierarchy represented as a Hesse diagram that classifies some diuretic drugs according to the Anatomical Therapeutic Chemical (ATC) Classification System (fDSM). In this example, *C03AA Thiazides, plain* is a subconcept of *C03 DIURETICS*, and *C03BA Sulfonamides, plain* is a superconcept of *C03BA04 chlortalidone*. Therefore a plain thiazide *is a* diuretic drug, and chlortalidone *is a* plain sulfonamide drug.

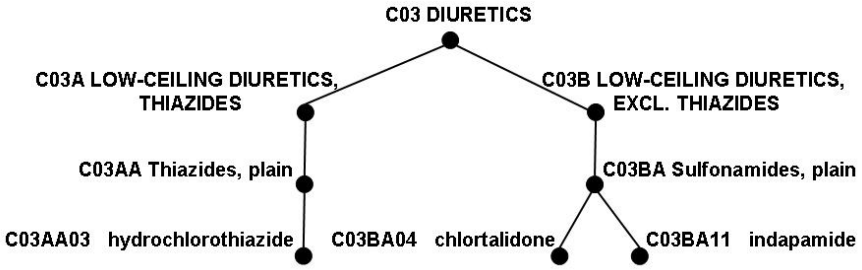


Figure 2.6: Example of concept hierarchy

Concept hierarchies may be very large and it can be difficult to visualize them using a Hesse diagram. Therefore, we can choose a tabular representation. The previous hierarchy is represented in a tabular way in table 2.4.

Table 2.4: Example of concept hierarchy represented as a table

C03 DIURETICS
C03A LOW-CEILING DIURETICS, THIAZIDES
C03AA Thiazides, plain
C03AA03 hydrochlorothiazide
C03B LOW-CEILING DIURETICS, EXCL. THIAZIDES
C03BA Sulfonamides, plain
C03BA04 chlortalidone
C03BA11 indapamide

Definition 2.5.15 (Nearest ancestor) *A concept y is called the nearest ancestor of concept x if $x, y \in H$ and $x \leq y, x \neq y$, and there is no other concept $z \in H$ such that $x \leq z$ and $z \leq y$ (i.e, x is a lower cover of y).*

In the example displayed in table 2.4, *C03AA Thiazides, plain* is the nearest ancestor of *C03AA03 hydrochlorothiazide*.

Definition 2.5.16 (Regular concept hierarchy) *A concept hierarchy* $\mathcal{H} = (H, \leq)$ *is regular if there is a greatest element in* H *and there are* n *sets* $H_1, H_2, ..., H_n$ *such that,*

$$H = \bigcup_{l=1}^{n} H_l \text{ and } H_i \cap H_j = \emptyset \text{ for } i \neq j$$

and, if a nearest ancestor of a concept in H_i *is in* H_j*, then the nearest ancestors of the other concepts in* H_i *are all in* H_j*.*

The concept hierarchy in table 2.4 is regular.

Another important term for describing the degree of generality of concepts is the level. Levels are assigned a number. We assign zero to the level of the greatest element (called the most general concept) of H, and the level of each other concept is assigned one plus its nearest ancestor's level number.

Due to the layered structure of a hierarchy, we notice that all the concepts with the same level number must be in set H_l for one and only one $l, l = 1, ..., n$. We thus simply call H_l as level l of the concept hierarchy.

The Hasse diagram of a concept hierarchy is actually a tree. Therefore, all the terminology for a tree such as node, root, path, leaf, parent, child, sibling, etc. is applicable to the concept hierarchy as well.

Definition 2.5.17 (Level name) *A level name is a semantic indicator assigned to a particular level. We denote* S *the set of level names of a concept hierarchy* $\mathcal{H}$*.*

Definition 2.5.18 (Schema level order) *A schema level order of a concept hierarchy* $\mathcal{H}$ *is a total order* $<$ *on* S *such that* $a < b$ *if there are two concepts* x *and* y *such that* x *is in* H_i *whose level name is* a *and* y *is in* H_j *whose level name is* b*.*

For example, in figure 2.6 we have the schema level order specified by the ATC classification system: *therapeutic group* $<$ *pharmacological subgroup* $<$ *chemical group* $<$ *active principle*, where *therapeutic group* is the name of the level containing the concept *C03 DIURETICS*; *pharmacological subgroup* is the name of the level containing the concepts *C03A LOW-CEILING DIURETICS, THIAZIDES* and *C03B LOW-CEILING DIURETICS, EXCL. THIAZIDES*; *chemical group*, the name of the level

containing the concepts *C03AA Thiazides, plain* and *C03BA Sulfonamides, plain*; and *active principle* the name of the level containing the concepts *C03AA03 hydrochlorothiazide*, *C03BA04 chlortalidone* and *C03BA11 indapamide*.

Definition 2.5.19 (Nearest common ancestor) *A concept z is the nearest common ancestor of two concepts x and y in a concept hierarchy $\mathcal{H}$ if and only if $z \leq x$ and $z \leq y$ and there is not any concept z' such that $z \leq z' \leq x$ and $z \leq z' \leq y$.*

In the previous example, *C03 DIURETICS* is the nearest common ancestor of *C03AA Thiazides, plain* and *C03BA11 indapamide*.

In medicine, several approaches have been done on the automatic induction of concept hierarchies (SWW94), on the automatic classification according to a concept hierarchy (SAM05; MCKD$^+$06) or on the use of concept hierarchies as background knowledge (LWY04; LVRC12b). It is common to use concept hierarchies of a standard classification system or terminology. Some of the most typical systems containing hierarchies of concepts which can be used as background knowledge are the Unified Medical Language System (UMLS) which is a compendium of many controlled vocabularies in the biomedical sciences (BBB$^+$96), the SNOMED CT (Systematized Nomenclature of Medicine – Clinical Terms) which is a systematically organized computer processable collection of medical terminology covering most areas of clinical information such as diseases, findings, procedures, microorganisms, pharmaceuticals etc. (LNM$^+$09) and the Anatomical Therapeutic Chemical (ATC) Classification System which is used for the classification of drugs (GKHAF09).

2.5.1.4 Other background knowledge structures

Several other mechanisms are used to represent background knowledge as cost functions, ontologies or co-occurrences.

Cost functions are used to give cost values to the different elements of a set according to a certain criterion. They are used in optimization problems, where we want to find the element in the set that optimizes the cost function.

Definition 2.5.20 (Cost function) *A cost function f is defined as a function $f : S \to \mathcal{D}$ that represents a certain criterion. The elements $s \in S$ are evaluated by f which gives them a cost value in the range $\mathcal{D}$ according to the criterion represented. Cost*

functions are used in optimization problems where the objective is to find the element in S that optimizes the cost function f.

The criterion represented by a cost function may be, for example, an economic cost, a duration, a level of health-risk, or a combination of several of them. According to the criterion, the range $\mathcal{D}$ of the function can be, for example, a natural number representing monetary units (euros), temporal units (seconds), abstract units of utility (utils), etc. Usually, in cases where the cost function has a maximum value, this range is normalized obtaining a cost function of the type $f : S \to 0..1$ using equation 2.6.

$$f_{norm}(x) = \frac{f(x)}{max_{s \in S} f(s)} \tag{2.6}$$

Cost functions are given a negative sense, so the higher the cost value of a certain element is, the worse this element is according to the criterion represented. A cost function f representing health risk is in the correct sense since a high level of risk has a negative sense. On the contrary, a cost function f representing patient comfortability goes in the opposite sense since a high level of comfortability has a positive sense. In these cases, we talk of benefit instead of cost. The corresponding function f' representing the cost on comfortability should be $f' = 1 - f$

In medicine, cost functions have been used to represent different kinds of criteria such as economic costs (LYWZ04; SY06), health risk (FCPB07) or several other medical criteria as comfortability or medical adherence (LVRB12).

Ontologies are structural frameworks for organizing knowledge and they are often used to formally represent complex background knowledge to share.

Definition 2.5.21 (Ontology) *An ontology is defined as a structural framework that represents knowledge as a set of concepts within a domain, and the relationships between those concepts. Regardless of the language in which they are expressed, most ontologies describe individuals (instances or ground level objects), classes (concepts), attributes that individuals and classes may have, and relations in which individuals and classes can be related to one another.*

Ontologies have been widely used to formalize and standardize medical knowledge. For example, MeSH is the National Library of Medicine's controlled vocabulary thesaurus which consists of sets of terms naming descriptors in a hierarchical structure

that permits searching at various levels of specificity. It contains thousands of descriptors which are arranged in both an alphabetic and a hierarchical structure and with cross-references. Another example is the metathesaurus of UMLS which contains over 1 million biomedical concepts (definitions) and 5 million concept names from more than 100 controlled vocabularies like ICD-10, MeSH or SNOMED CT used in patient records, administrative data, full-text databases and expert systems.

Some examples of the use of ontologies as background knowledge in medicine are (FYL+06) and (TBK09) where Gene Ontology annotations are used in bioinformatics to assist a clustering process, and (RRLV+12) that presents an ontology-based system that helps health care professionals to determine patient conditions and also to provide a coordinated action plan which is adapted to the patient needs.

Another type of background knowledge specifies which combinations of values (co-occurrences) of a set of attributes have high importance for a classification problem.

Definition 2.5.22 (Co-occurrence) *A co-occurrence (or typical co-occurrence) is defined as a combination of values of a set (grouping) of attributes that represent a characteristic combination.*

For example, co-occurrences have been used as background knowledge in (ZD98) where the possibility of automating the process of acquiring co-occurrences is explored and studied in the problem domain of rheumatic diseases.

2.5.2 Background knowledge in clustering

Clustering with background knowledge is also usually referred to as *semi-supervised clustering*. We identify four approaches of semi-supervised clustering: constrained clustering, seeded clustering, metric-based clustering and rule-based clustering.

In constrained clustering, background knowledge is represented as constraints that must be respected during the clustering process. The most typical kind of constraints are pairwise must-links and cannot-links. A must-link constraint between two instances means that these two instances should be in the same cluster and a cannot-link means that they should not be in the same cluster. In (WC10) the COP-COBWEB is presented as a modification of the COBWEB clustering algorithm including must and cannot-links. In (WCRS01) the same procedure is followed with the k-means clustering algorithm obtaining the so-called constrained k-means (COP-kmeans). An evolved version of

these constraints is used in (KKM02) to modify a hierarchical agglomerative clustering algorithm. Another approach based in k-means, uses these kind of constraints with an associated cost of violating each constraint (BBM$^+$04a). Must-link constraints are used to cluster documents in (JX06).

Seeded clustering uses some labeled instances (instances that have been previously assigned to a cluster) as background knowledge. A k-means based approach uses seed instances in (BBM02; SKP03). In (BBM04b) a hierarchical agglomerative clustering algorithm uses seed clusters.

In metric-based approaches, an existing clustering algorithm that uses a distance metric is employed; however, the metric is first trained to satisfy the labels or constraints in the supervised data. (KKM02) uses an Euclidean distance trained by a shortest-path algorithm, (BM03) uses string-edit distance learned using Expectation Maximization (EM), (CCM03) adapts KL divergence with gradient descent and (XNJR03) uses Mahalanobis distances trained using convex optimization.

Rule-based clustering represents background knowledge by means of rules that guide the learning process. The ISAAC algorithm is modified using classification rules in (TB99). (VBL09) uses conjunctive rules to introduce additional background knowledge to the ClusDM methodology.

Finally, some approaches combine several of the previous methods. In (BBM04c) the k-means algorithm is modified (MPCK-means) using an approach that uses both constraints and metric learning and (KK08) uses supervision in terms of relative comparisons (e.g., x is closer to y than to z) and also learns the underlying dissimilarity measure.

2.5.3 Background knowledge in decision making

The most common way of including background knowledge in the induction of decision mechanisms is by means of cost functions. This approach is known as *cost-sensitive learning*. In the case of DTs, they are referred to as cost-sensitive decision trees.

The costs considered in cost-sensitive learning may be of different kinds (Tur00) as for example, cost of misclassification errors or cost of tests.

The *cost of misclassification errors* measures the error of making a wrong decision. For example, the error of sending a patient home when it should have been sent to ICU. It is usually represented by a cost function $e(D, D)$ where D is the set of possible

decisions. The error $e(d_i, d_j)$ specifies the cost of making a decision d_i over an instance, when it is the correct decision is d_j. In (LVRB12) a different approach is presented using the concepts of type I and type II errors. Type I error represents the relevance of not making a correct decision (e.g., not accepting a patient to an ICU when it is required) and type II error represents the relevance of making a wrong decision (e.g., discharging a patient when it should remain at the hospital). With this approach we have two cost functions $e_I(D)$ and $e_{II}(D)$ representing type I and type II error respectively. Most approaches deal with constant error costs (i.e., the same value for all instances) but costs may be conditional (i.e., they may depend on the nature of the particular instance, the timing, whether errors have been made with other instances or the answer of one or more questions of the instances). Some reviews of misclassification costs are (Elk01; GGR02). In medicine, (LYWZ04; SY06) considers economic costs and (LVRB12) classifies the most important criteria in medical decision making and includes some of them as misclassification costs.

The *cost of tests* measures the cost of obtaining the answer to a certain question. For example, the cost of answering a question that implies a surgical test may be greater than the cost of a question that can be answered with a non-invasive test. It is usually represented by a cost function $e(Q)$ where Q is the set of possible questions. The error $e(q_i)$ specifies the cost of answering question q_i for an instance. In these cases, costs can also be constant or conditional. In the case of conditional costs, they may depend on the previous questions that have been asked, the answer of the previous questions, the possible side-effects of the test, etc. In the bibliography, some approaches use cost of tests usually making a trade-off between a cost function and the typical information gain measure (Nor89; Tan93). In medicine, (LYWZ04; SY06) considers economic costs, (FCPB07) also includes the health-risk criterion and (LVRB12) uses several other medical criteria as comfortability or medical adherence.

There are other kinds of cost which have been less considered in the bibliography like cost of teacher, cost of intervention, cost of unwanted achievements, cost of computation, cost of cases, human-computer interaction cost or cost of instability (Tur00).

In spite of cost-sensitive learning, there are several other ways to include background knowledge in decision making. For instance, (Nuñ91) combines cost functions with a IS-A hierarchy and (Tin98) weights the instances of the dataset. In the health care domain, (LVRC07) represents the adherence to the medical guidelines as a partial order

over the questions and makes a trade-off with the information gain. Partial orders are also used in (LVRB12) to represent medical criteria.

2.6 Incrementality

Incrementality is defined as the process of increasing in number, size, quantity, or extent. In machine learning, *incrementality* refers to the property of being able to deal with new incoming data to revise, if necessary, a previously induced mechanism without re-inducing it from scratch. In (GC00) Giraud-Carrier states that "a learning task is incremental if the training examples used to solve become available over time, usually one at a time".

Incremental induction is desirable for a number of reasons (Utg94). Most importantly, revision of existing knowledge presumably underlies many human learning processes, such as assimilation and generalization. Secondly, knowledge revision is typically much less expensive than knowledge creation. For example, incrementality is useful for serial learning tasks, on the assumption that it is more efficient to revise an existing hypothesis than it is to generate a hypothesis each time a new instance is observed. Finally, the ability to revise knowledge in an efficient manner opens new possibilities for algorithms that otherwise would remain prohibitively expensive.

Incremental learning algorithms are usually motivated basically for these three desirable goals:

1. Cost reduction: The incremental cost of updating the current hypothesis with a new instance should be much lower than the cost of building a new hypothesis from scratch. It is not necessary however that the sum of the incremental costs be less than the execution on the complete database.

2. Independence from the size: The update cost should have a high degree of independence to the number of training instances on which the decision mechanism is based.

3. Independence from the order: The hypothesis produced by the incremental algorithm should depend only on the set of instances that has been used, without regard to the sequence in which these instance were presented.

2.6.1 Incremental clustering

The interest in incremental clustering stems from the fact that the main memory usage is minimal since there is no need to keep in memory the mutual distances between instances and the algorithms are scalable with respect to the size of the set of instances and the number of attributes.

One of the most famous incremental clustering algorithms is COBWEB (Fis87). It maximizes a measure called category utility to build a probabilistic hierarchical tree. The algorithm reads one instance per iteration from a data set and incorporates it into the tree by descending the tree along an appropriate path to a node where the category utility is maximal after absorbing the instance and updating statistical information (for computation of the probabilities) in each node along the way. To find the proper place to hold the instance, COBWEB tries one, or several, or all of the following four possible operations at each node on the path:

1. place the instance in an existing cluster
2. create a new cluster by itself
3. merge the best two clusters with respect to the values of category utility
4. split a cluster into several clusters by lifting its children one level in the tree to replace itself

The operation resulting in the largest value of category utility is the final choice on that node. This procedure is recursively invoked until a leaf node is reached or a new leaf is created.

Another approach is proposed in (CCFM97) where the number of clusters is fixed. When a new instance arrives, either it is assigned to an existing cluster or a new cluster is created while two of the existing clusters are merged. A method based on this approach is used in (SH01) for document clustering. In (EKS$^+$98) the DBSCAN clustering algorithm is adapted to deal with incremental data. The incremental algorithm for nominal data in (SSK04) is based on a metric on the set of partitions of a finite set of instances. Other alternative approaches are based on swarm intelligence (LPM06) or neural networks (HBBC08).

In some domains, it is possible that not only the data set evolves with the incorporation of new instances, but the set of attributes may be dynamic. In (cC05) an incremental clustering algorithm Core Based Incremental Clustering (CBIC), based on k-means, is presented which is capable to construct a new partition of the data set, when the attributes set increases.

In medicine, incremental clustering has been applied in several applications, for example to detect of infectious outbreaks in hospitals (LGCM01) or in the cancer domain (dV96).

2.6.2 Incremental decision making

Most of research on incremental decision making has been based on improvements of the ID3 DT induction algorithm (Qui86) both finding a way to compute the information gain with minimal spatial cost and guaranteeing that the DTs incrementally obtained are equal to those that would generate the non-incremental ID3. The first approach proposed is the ID4 algorithm (SF86) which follows a ID3 based algorithm and when the relative ordering of the possible questions at a node changes due to new incoming instances, all subtrees below that node are discarded and have to be reconstructed, causing that certain concepts are unlearnable. The minimal information needed to compute the information gain for a possible question at a node is kept. This information consists of positive and negative counts for each possible answer to each possible question at each node. The ID4 algorithm builds the same tree as the basic ID3 algorithm only when there is a question at each decision node that is clearly the best choice in terms of its information gain. The ID5 algorithm (Utg88) does not discard subtrees, but also cannot guarantee that it will produce the same tree as ID3. The ID5R algorithm (Utg89) produces the same tree as ID3 for a data set regardless of the incremental training order. This is accomplished by recursively updating the tree's subnodes. ID5R restructures the tree so that the desired question is at the root. The restructuring process, called a pull-up, is a tree manipulation that preserves consistency with the observed training instances, and that brings the indicated question to the root node of the tree or subtree. The advantage of restructuring the tree is that it allows recalculating the various positive and negative counts during the tree manipulations, without reexamining the training instances. The improved ITI algorithm (UBC97) also

produces the same tree regardless of the presentation order, or whether the tree is induced incrementally or non incrementally (batch mode). It can accommodate numeric variables, multiclass tasks, and missing values.

In spite of ID3 based incremental algorithms, other approaches are the incremental CART algorithm (Cra89) which is based on the non-incremental CART algorithm (BFOS84), the STAGGER algorithm (SRG86) which examines concepts that change over time (concept drift) or the Very Fast Decision Trees learner (VFDT) (DH00) which reduces training time for large incremental data sets by subsampling the incoming data stream.

In medicine, incremental decision making has been applied in the retrieval from manuals and medical texts (WS06), for on-line prediction of hospital resource utilization (NML06) or for patient-dependent seizure detection (Wil05) among others.

2.7 The EOC Data Model

Health care deals with the concept of *encounter* between the patient and the health care professionals (RLVT08). An *Episode Of Care (EOC)* of a particular patient is the sequence of encounters aiming at curing, stabilizing, or palliating one or several of that patient's ailments. Notice that chronic diseases define EOCs that remain open for the patient's entire life. Concerning a single encounter, the standard behavior of a health care professional is to observe the current state of the patient (e.g., patient symptoms, test results, etc.) and then decide some actions (e.g., prescribe drugs, order tests, start some medical procedure, etc.). Observe that some evidence may exist that justify these actions. Therefore within the same encounter, several health care measures may coexist containing, each one, the evidence to a subset of the actions performed during that encounter. For example, in the hypertension domain, for a particular encounter the physician may decide both a drug therapy based on the evidence that the patient is at high risk of cardiac disease, and a recommendation to modify the patient lifestyle, due to the presence of cholesterol. A representation model for this minimal information about the treatment of a chronic patient is the *EOC data model* (Ria10). A simplified formalization of it can be seen in table 2.5.

Notice that the data about patient condition, evidences and actions is represented by means of terms in order to fit the terminology used in the SDA knowledge representation

Table 2.5: Simplified formal description of the EOC data model

episode of care	← sequence of encounters
encounter	← patient condition + list of health care measures
patient condition	← list of state terms
health care measure	← evidence + action
evidence	← list of decision terms
action	← list of action terms

model (see section 2.8). These terms (or vocabulary items) can be of three sorts: state terms, decision terms and action terms and they are detailed in the next section. Let enc_i be an encounter in the EOC database, we call $S(enc_i)$ the set of state terms that define the patient condition in enc_i, and $D(enc_i)$ and $A(enc_i)$ the set of decision terms and action terms contained in each health care measure in enc_i.

2.8 The SDA knowledge model

The State Decision Action (SDA) knowledge model (Ria07; BRLV12) is used to represent procedural knowledge in medicine stressing the concept of simplicity without losing description capability. It is based on the concept of Clinical Algorithms (CAs) (SfMDM92; Had95) but also includes all the representation primitives that any CIG (Computer-interpretable guidelines) system is expected to have (PPT$^+$02; PTB$^+$03; MvdAP07; IM08) (i.e., actions, decision, patient states, execution states, sequences, concurrences, alternatives, and loops). The SDA knowledge model is founded on the concept of *term* or vocabulary item in the medical domain of the procedural knowledge. These terms can be of the sort state, decision, or action. *State terms* define the vocabulary that is used to describe the feasible patient conditions and situations in the area of interest (e.g., terms as Elevated_Blood_Pressure or Following_Drug_Treatment to establish a differential treatment). *Decision terms* are the terminology that health care professionals use to condition the sort of treatment to be followed (e.g., terms as Secondary_Cause_Suspected or BP_at_Goal that may derive the course of professional activities in one direction or another). *Action terms* are the way that medical, surgical, clinical or management activities are defined (e.g., terms as LifeStyle_Modifications or Drug_Therapy are respective examples of counsel and prescription, which are two of the types of medical actions that may appear in the description of a treatment. In a certain medical context, we denote S, D and A, the sets of state, decision and action

terms respectively. State, decision and action terms are employed to construct three sorts of elements that once interconnected they will describe the medical procedure. These elements are, respectively: states, decisions and actions. *States*, which are subsets $S_i = \{s_{i_1}, s_{i_2}, ..., s_{i_n}\}$ where all s_x are state terms, represent patient conditions, situations, or statuses that deserve a particular course of action which is totally or partially different from the actions followed when the patient is in another state, for example, to differentiate between the initial treatment and the subsequent treatments or between the different stages of a disease. *Decisions* allow the integration of all the variability that a treatment may have by means of conditions on several decision terms $d_{i_1}, d_{i_2}, ..., d_{i_m}$ which represent some of the available information about the patient and the current situation. *Actions*, which are subsets $A_i = \{a_{i_1}, a_{i_2}, ..., a_{i_p}\}$ where all a_x are action terms, constitute the proper health care activities involved in the health care procedure represented. Similar to the CA notation, the SDA model represents states, decisions, and actions respectively as circles, diamonds, and rectangles which are connected with arrows in order to provide a join representation of a health care procedure, as the one depicted in figure 2.7 for the treatment of hypertension. It distinguishes between plain connectors, decisional connectors, and otherwise connectors. *Plain connectors* represent evolutions of the health care procedure which can be followed by any patient. *Decisional connectors* link decisions with other elements, they contain decision terms, and only the patients who meet all the terms in a connector are able to follow this connector. Finally, *otherwise connectors* link decisions with other elements, they are identified with the word 'otherwise', and only the patients who fulfill none of the connectors leaving a decision are able to follow the otherwise connectors of that decision. The sequence of decisions and actions that connects one state to one or more next states is called *therapeutic sequence*. See, for example the SDA in figure 2.7.

Connectors may have time constraints of the form [min, max]; *min* representing the minimum time the process must stop before following the connector (e.g., wait two hours before measuring BP again to confirm high BP), and *max* the maximum time the process must stop before moving forward in the treatment (e.g., next visit must be scheduled for not later than one week).

The interpretation of a SDA diagram is the following: when a patient arrives, all the SDA states whose state terms are observed in the current patient condition are eligible to start the treatment. If several states are eligible, a health care professional

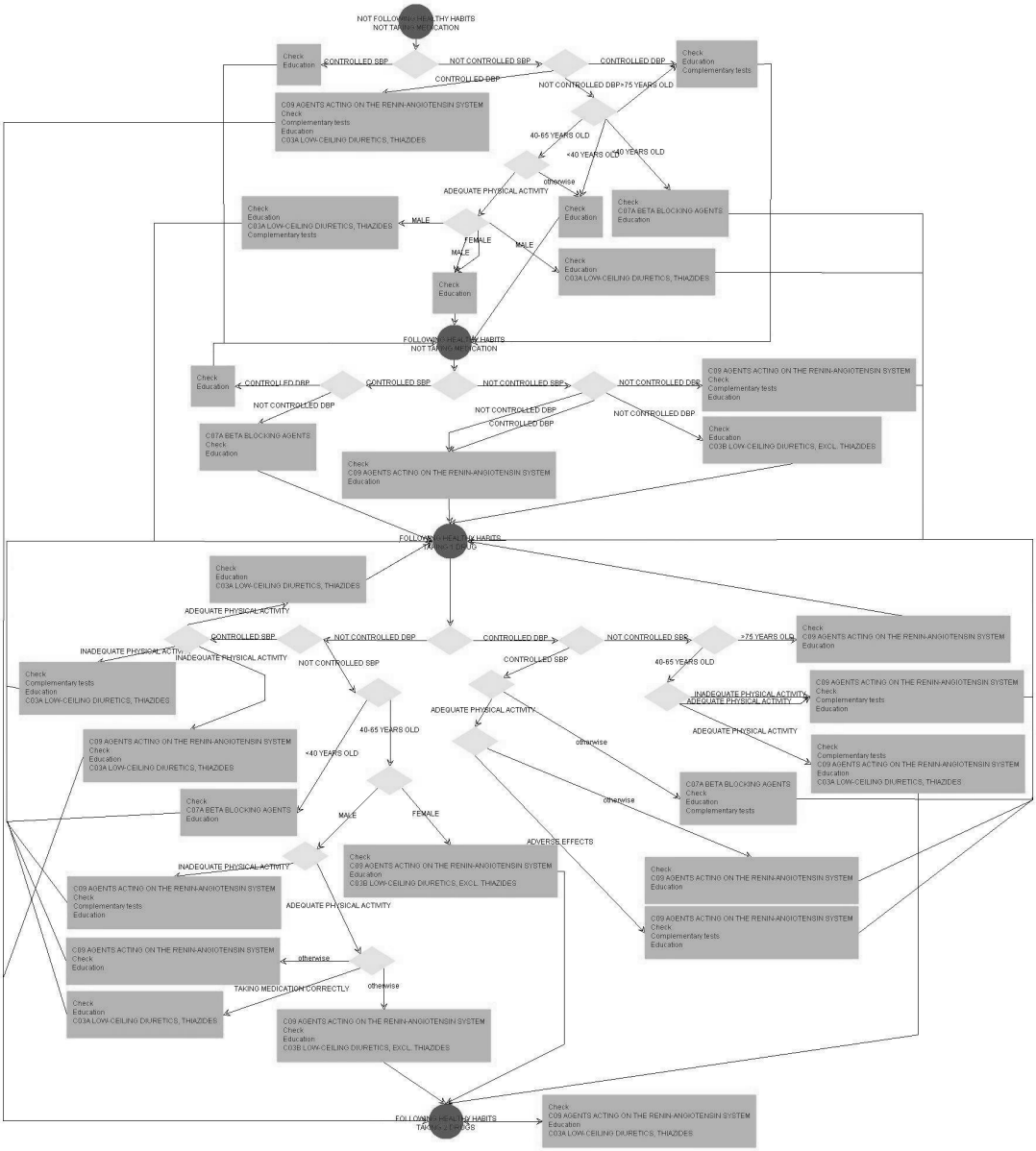


Figure 2.7: Example of SDA diagram for the treatment of hypertension

has to decide the one to start at among all the eligible states (this is called *type-0 non-determinism*). Once this is decided, the connectors are followed until either a non-eligible state is found or a connector with a positive min delay is reached. In this process, all the actions of the followed path are the SDA recommendations for the treatment of that patient. When a decision is reached, all the outgoing decision connectors whose decision terms are part of the patient condition are eligible to follow the treatment of that patient. If only one decision connector is eligible, the connector is followed. If there are several eligible connectors, then a health care professional has to choose one of them to follow the treatment (this is called *type-1 non-determinism*). If none of them is eligible, but there is an otherwise connector, then this connector is followed. If several otherwise connectors exist, then a health care professional decides which one is the one to be followed (this is also considered type-1 non-determinism). In case that there are several plain connectors leaving a state or an action, all of them are eligible and it is the health care professional who has to decide the one to be followed (this is called *type-2 non-determinism*). Non-determinism is only observed when there is not a single accepted and evidence-based procedure to deal with a particular situation and the choice criterion between the alternatives is not defined.

The SDA model has been thoroughly tested in the context of the K4CARE project (www.k4care.net) where it has been successfully used to represent different sorts of procedural knowledge in medicine, particularly in home care.

2.9 Induction of medical procedural knowledge

Diffusion of Information and Communication Technology tools within the health care practice, such as electronic clinical charts, computerized guidelines and, more generally, decision support systems, makes a huge amount of data available which can be exploited. One of the main uses of this data is the *induction of medical procedural knowledge* which consists in mining the hospital databases by means of machine learning techniques in order to obtain structures that represent the different flowcharts followed by patients with a certain disease or pathology.

This kind of structures have been widely used in health care for ages. This is the case of CAs (SfMDM92; Had95) which are included in Clinical Practice Guidelines (CPGs) to graphically summarize some of the medical procedures described in the guideline.

The CAs defined by the international Society for Medical Decision Making (SfMDM92) are flowcharts that start with a clinical state box defining the clinical state or problem, and then a combination of both, decision boxes representing yes-no questions leading the process to alternative paths, and action boxes describing actions, either therapeutic or diagnostic. All these boxes are connected by arrows that show the logical sequence of application of the CA. For example, the CA in figure 2.8 was published by the Institute for Clinical Systems Improvement (Sch06) as a generalization of the long term treatment and follow up of hypertension.

Generally, CAs are hand-made which represents a laborious task that implies the interaction between several health care experts of different specialties. This is not the only drawback of the manual creation of CAs. The individual differences of the patients causes great variances in the application of CAs in daily practice. In real world, chronic patients use to suffer of more than one disease (comorbidities) and each case has some particularities that may not be considered by the CA. The induction of structures like CAs from hospital databases and medical resources solves the previous drawbacks. It reduces the high costs of the manual generation and it allows the analysis of health care in comorbidities. Moreover, the automatic induction of CAs can be used to:

- Create different views (or dimensions) about the same clinical activity (e.g. only pharmacological treatment, only nursing activities, only expensive actions, patient evolution vs administrative issues, etc.) by the selection of different terminologies to express the induced CAs
- Complement and refine the CAs based on the medical knowledge provided by the CPGs, because the induced CAs are obtained from clinical experience
- Automatically check guideline compliance
- Increase the understanding of disease processes
- Improve the physician education
- Compare the procedures followed by different health care institutions

One of the main approaches in the induction of medical procedural knowledge from data is based on the so-called workflow mining (vdAvDH$^+$03). This technique generates

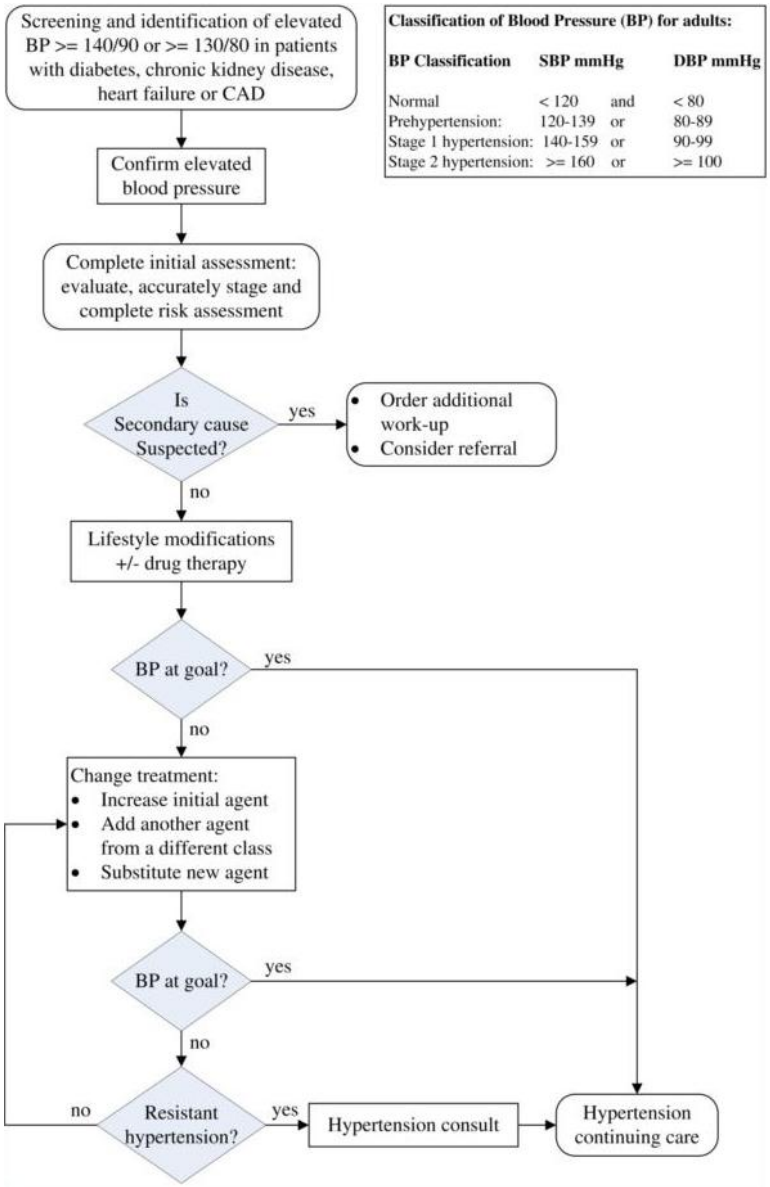


Figure 2.8: Clinical algorithm on hypertension published by the Institute for Clinical Systems Improvement

process related information exploiting the event logs from a process management system, database, etc. and has been successfully applied in several domains (vdAvDH$^+$03). In (MSSvdA08) and (MSL$^+$08) workflow mining is applied in gynecological oncology and stroke respectively obtaining clinical pathways represented as Petri nets. The main drawback of this approach is that the structures induced by those systems are not explicit medical structures that doctors are as familiar to work with as with CAs.

Another approach is the induction of SDA diagrams using machine learning techniques (RLVT08; BRLV12). The SDA knowledge model (Ria07; BRLV12) has been introduced in section 2.8 as a model to represent medical procedural knowledge which is similar to CAs with some improvements. Firstly, the presence of states for the different stages of a certain disease or disorder lets the SDA model to depict several treatments in an integrated diagram allowing the representation of long term procedures. Another improvement is that it can deal with multiple entry points corresponding to the states that represent the different initial patient conditions and, therefore, not only to integrate the treatment of all these conditions in a single diagram, but also to address each patient directly to the corresponding part of the treatment. SDA diagrams also extend the expressiveness of CAs using multi-term decisions. In CAs, decisions are always (SfMDM92) yes-no questions but, in the SDA model, decisions may have more than two branches with different decision terms in each one of them. In addition, each decision may have alternative otherwise branches which are followed by the patients that fulfill none of the other branches. This results in a more readable sequence of decisions and also in a more compact representation of treatments. Finally, the rigidity and strictness of CAs, previously referred to as their main criticism, is reduced in the SDA model which increases the flexibility of CAs by dealing with non-determinism. Non-determinism is frequent in medicine and it allows the participation of health care professionals when there is not proven evidence on a unique or better treatment.

The current approach to induce SDA diagrams (BRLV12) can be summarized in 4 stages:

1. Detecting states: The states of the SDA diagram are detected with a method based on a syntactic similarity function between states.

2. Detecting actions: The different sorts of actions of the SDA diagram are detected with a method based on a syntactic similarity function between actions.

3. Determining evolutions: The sequences of decisions corresponding to the evolution from each state to each other possible state are determined using induction of decision trees.

4. Determining actions: For each evolution between states obtained in the previous stage, the sequences of decisions corresponding to each possible action are determined using induction of decision trees.

5. Integrating: The different states and sequences of decisions and actions are integrated in the final SDA diagram.

Notice that this approach does not consider any kind of background knowledge of the domain to guarantee medically correct results and it uses a non-incremental algorithm. However, it achieved successful results in (RLVT08) where it was applied to hypertension, cervical cancer, colorectal cancer and chronic obstructive pulmonary disease. Two sorts of test were performed: one oriented to verify if the algorithm was able to recover a predefined SDA diagram from a representative sample of patients treated according to the indications of that SDA diagram, and another one centered on the generation of a SDA diagram from the medical actions recorded in a certain hospital. In (BRLV12) the methodology was tested on the medical domain of hypertension with the purpose of studying the differences between the health care procedures of a hospital database and some predefined official CAs.

3

Medical background knowledge

In the automatic generation of medical decision structures, there is a relevant amount of *background knowledge* which is not explicitly included in the input data. The exclusive use of mathematical or statistical measures to induce structures that summarize the steps registered in a hospital database has been proved to be not successful because it leads to results which are not medically correct or which health care professionals may not be familiar with (LVRC07; LVRB12). In a complex and sensitive domain like medicine, using measures that consider all the background knowledge involved is essential in order to assure medically correct results. This kind of knowledge consists of medical criteria and constraints that health care experts take into account during their medical activities. It can be provided by the health care experts themselves or it can be extracted from medical knowledge sources. This background knowledge is of different kinds and each one is *formalized* with a different knowledge structure.

As we stated in the introduction, one of the objectives of this thesis is to build a repository of background knowledge corresponding to the diseases of Hypertension (HT) (ohhs03), Diabetes Mellitus (DM) (CL08) and the comorbidity of both diseases (MSRS07). By means of a knowledge engineering process we have obtained this *repository of knowledge structures*.

3.1 Formalization of medical background knowledge

The automatic induction of medical decision structures involves several kinds of knowledge. From all the possible kinds of background knowledge in health care we will focus

on the constraints that affect the desired set of possible health care states, the preferences about the terminology used in health care states, the semantic relationships between decision terms, the order in which some questions should be done and the similarity between different medical actions. Each of these kinds of knowledge must be represented with a certain background knowledge structure. Here we present these different kinds of medical knowledge needed and the knowledge structures that are used to formalize them.

3.1.1 Constraints on health care states

The treatment of a pathology usually deals with the so-called *health care states* which are used to organize the different patients according to their conditions.

Definition 3.1.1 (Health care state) *A health care state is a set of patient conditions, situations, or statuses which involve a significant group of patients that deserve a particular course of action which is totally or partially different from the actions followed when the patient is in another health care state and which has some interest for the health care professional.*

A health care state must have a medical sense that depends on both the coherence of its description and the coherence of the health care state itself with respect to the rest of health care states. That is to say that given a description of a health care state it must not be redundant or medically incorrect, and given the whole set of health care states they must be defined at the same level of abstraction of the medical terminology. This medical sense of a health care state is a kind of medical knowledge which consists in constraints in the description of health care states.

In the SDA model, health care states are represented as SDA states which are subsets of SDA state terms (see section 2.8). The set of state terms within a SDA state define its description, so the constraints related to the description of the health care states are constraints between state terms, and they are called *state term constraints.*

Definition 3.1.2 (State term constraint) *Being S a set of state terms, a state term constraint c is an unordered pair $\{s_i, s_j\}$ with $s_i, s_j \in S$, meaning that if a SDA diagram includes a state S_i that contains the term s_i, then s_j will not be allowed in any of the states of the SDA diagram (including S_i).*

We represent these constraints by means of a *state constraints graph*.

Definition 3.1.3 (State constraints graph) *A state constraints graph is defined as an undirected graph $G_S = (S, C)$ (see section 2.5.1.1) where S is a set of state terms and C is a set of state term constraints between state terms in S.*

Table 3.1 contains a state constraints graph for the treatment of hypertension. For example, if a state is included in the SDA diagram such that it contains the term *HEART RISK*, then none of the states of the SDA diagram will contain the term < 40 *YEARS OLD*.

Table 3.1: State constraints graph for HT (a part of)

	HEART RISK	NOT HEART RISK	< 40 YEARS OLD	40 .. 65 YEARS OLD	65 .. 75 YEARS OLD	> 75 YEARS OLD
HEART RISK			X	X	X	X
NOT HEART RISK			X	X	X	X
< 40 YEARS OLD						
40 .. 65 YEARS OLD						
65 .. 75 YEARS OLD						
> 75 YEARS OLD						

3.1.2 Preference between state terms

Considering all the possible health care states that can be used to classify patients during the treatment of a pathology, a health care professional may have preference for some of them depending on the context. For example, for the treatment of hypertension a general practitioner could prefer organizing the patients according to the stages of the disease while a hospital administrative assistant could be more interested in the units or departments where the patients are being treated. This kind of knowledge is a preference order between the possible health care states.

Therefore a SDA diagram (see section 2.8) is constructed for a concrete purpose (e.g., visualizing the different stages of a treatment, representing actions in emergence situations, supporting the selection of drugs, prevention, etc.). The construction of a SDA diagram is associated to an intentionality which indicates the context in which the SDA diagram will be used. The set of SDA states is used to express this intentionality, so a health care professional may have preference for some state terms depending on the different states that he wants in the final diagram. We represent these preferences by means of a *state terms partial order*.

Definition 3.1.4 (State terms partial order) *A state terms partial order $\leq_S$ is defined as a Layered Partial Order (LPO) over the set of state terms S (see section 2.5.1.2) such that, given two terms $s_i, s_j \in S$, $s_i \leq_S s_j$ means that s_i is preferred than s_j.*

If $s_i \leq_S s_j$, the state term s_i will be more likely to appear in some of the states of the SDA diagram than s_j as it is preferred by the health care professional.

Table 3.2 contains a state terms partial order for the treatment of hypertension. States having *FOLLOWING HEALTHY HABITS* or *TAKING 3 DRUGS* will be more likely to be selected for the SDA diagram than those having *CONTROLLED DBP* or 65 .. 75 *YEARS OLD*.

Table 3.2: State terms partial order for HT (a part of)

Priority	State term
1	NOT FOLLOWING HEALTHY HABITS
	FOLLOWING HEALTHY HABITS
	NOT TAKING MEDICATION
	TAKING 1 DRUG
	TAKING 2 DRUGS
	TAKING 3 DRUGS
2	CONTROLLED DBP
	NOT CONTROLLED DBP
	CONTROLLED SBP
	NOT CONTROLLED SBP
3	$<$ 40 YEARS OLD
	40 .. 65 YEARS OLD
	65 .. 75 YEARS OLD
	$>$ 75 YEARS OLD
	LVH
	NO LVH

3.1.3 Semantic decisions

During an encounter with a patient, the physician determines some evidences in order to make some actions for the treatment of this patient. These evidences are obtained by asking questions, by consultations to the records or by performing health care tests. The sequence of gathering evidences and making a final action is called *therapeutic sequence*.

Definition 3.1.5 (Therapeutic sequence) *A therapeutic sequence is a tree-like sequence of evidences gathered by questions, consultations or tests that discriminate the different sorts of treatment to be followed by the patients evolving from a certain health care state*

to any other states. A therapeutic sequence may have no questions but it must have at least one treatment.

When the physician decides to ask a question to the patient, to consult his records or to perform a health care test he expects several possible outcomes (evidences). For example, a physician may want to determine the blood pressure of the patient. He knows that the expected outcomes when determining the blood pressure are low blood pressure, blood pressure at goal or high blood pressure. This is a kind of medical knowledge that relates these outcomes because they are the different alternative values of a same question, in this case, the blood pressure of the patient.

In the SDA model (see section 2.8), the separation of patients according to their medical evidences is made with SDA decisions. Decisions allow the integration of all the variability that a treatment may have by means of conditions on several decision terms. These decision terms are the different expected outcomes in a certain decision. When several decision terms can be the expected outcomes of a same decision, we say that they are *semantically related*. Following the previous example, we could have the decision terms Low_BP, BP_at_goal and $High\_BP$ which are expected outcomes when determining the blood pressure of the patient. These decision terms are semantically related because they represent a certain level of blood pressure. In a SDA diagram we can place a decision providing these three alternatives according to the blood pressure level of the patient. The possible alternatives when making a decision do not always have to be disjoint. In order to represent these semantic relationships as background knowledge, we use a semantic decisions hypergraph.

Definition 3.1.6 (Semantic decisions hypergraph) *A semantic decisions hypergraph is defined as a hypergraph $H_D = (D, SD)$ (see section 2.5.1.1) where D is the set of decision terms and SD is a set of hyperedges sd such that if $d_1, d_2 \in sd$ then d_1 and d_2 are semantically related.*

The semantic decisions hypergraph defines all the possible *semantic decisions*.

Definition 3.1.7 (Semantic decision) *A hyperedge $sd \in SD$ of a semantic decisions hypergraph H_D is called a semantic decision.*

All the decisions in a SDA diagram must represent semantic decisions (i.e., their decision terms must be semantically related).

Table 3.3 contains part of a semantic decisions hypergraph for the treatment of hypertension. For each semantic decision it specifies a name and the set of decision terms that it has. For example, the semantic decision called *BMI* is composed by the decision terms *NORMAL BMI*, *OVERWEIGHT BMI* and *OBESE BMI*. Therefore, considering these semantic decisions, the SDA diagram can include a decision with *NORMAL BMI*, *OVERWEIGHT BMI* and *OBESE BMI* as alternatives, but a decision with terms of different semantic decisions will not be allowed (e.g., *NORMAL BMI* and *TAKING MEDICATION CORRECTLY*).

Table 3.3: Semantic decisions hypergraph for HT (a part of)

SD name	SD terms
BMI	NORMAL BMI
	OVERWEIGHT BMI
	OBESE BMI
Cardiac auscultation	NORMAL CARDIAC AUSCULTATION
	NOT NORMAL CARDIAC AUSCULTATION
Correct medication	TAKING MEDICATION CORRECTLY
	NOT TAKING MEDICATION CORRECTLY

3.1.4 Order of decision sequences

The order in which the different needed evidences are gathered during an encounter in order to decide the treatment for a patient depends on several criteria that the physician may consider (LVR12). An important criterion is the utility of an evidence to decide the course of treatment of the patient but the physician may also consider the order specified in a clinical guideline, the risk on the health of the patient of obtaining this evidence, the uncomfortability caused to the patient or his own experience. Depending on one or more of these criteria, the physician will gather evidences in a certain order. If these evidences are gathered in a different order, the sequence followed may be not *comprehensible* by the physician. This kind of medical knowledge is an order between the evidences.

In the SDA model (see section 2.8) we use decisions to separate patients according to their evidences. The sequences of decisions must follow an order which is comprehensible for the health care expert. All the decisions in a SDA diagram must always be semantic decisions (see section 3.1.3), so the background knowledge about the order of decisions is represented with a partial order over the set of semantic decisions.

Concretely we use a layered partial order defined by the health care expert according to his own criteria called decisions partial order.

Definition 3.1.8 (Decisions partial order) *A decisions partial order $\leq_D$ is defined as a Layered Partial Order (LPO) over the set of semantic decisions SD (see section 2.5.1.2 and 3.1.3) such that, given the semantic decisions $sd_i, sd_j \in SD$, $sd_i \leq_D sd_j$ means that, according to the criteria of the health care expert, sd_i should be asked before s_j.*

If $sd_i \leq_D sd_j$, the SDA decision representing the semantic decision sd_i will be more likely to appear before sd_j in a therapeutic sequence.

Table 3.4 contains a decisions partial order for the treatment of hypertension. According to this partial order, the decisions about DBP or SBP will be more likely appear in the SDA diagram before asking for the age, the sex or the heart rate.

Table 3.4: Decisions partial order for HT (a part of)

Priority	**SD**
1	DBP
	SBP
2	Adverse effects
	Age
	Correct medication
	Physical activity
	Sex
3	Heart rate

3.1.5 Similarity between actions

The actions that can be made by a physician are of different kinds like pharmacological, educational, analytical, ECGs, radiological, consultation, verification, procedural, etc. For example, prescribing Prinivil 5mg or prescribing Enalapril Merck 2 mg is a pharmacological action. Although being syntactically different, two actions do not have to be semantically different. Depending on their medical meaning, two different actions can be similar or even equivalent. Following the previous example, these two drug prescriptions are syntactically different. However, both drugs are semantically similar because they are angiotensin converting enzyme (ACE) inhibitors which are used to prevent the blood pressure raising produced by the ACE. The physicians may

use different drugs interchangeably if they are similar enough. The knowledge about the similarities between actions consists in their semantical characteristics, the dosage equivalences, etc.

In the SDA model (see section 2.8) the different actions are represented with action terms. In the SDA diagrams, these terms are grouped in SDA actions. When inducing SDA diagrams, the knowledge about the similarities between actions terms and SDA actions is essential to solve problems like calculating the *homogeneity* of treatments within a set of encounters or when deciding the most appropriate SDA actions after a sequence of SDA decisions in order to obtain *correct* therapeutic sequences. We represent this knowledge using an extended concept hierarchy (see section 2.5.1.3) that contains the semantics of the action terms needed to calculate their similarities.

In the case of action terms representing pharmacological actions, the WHO has established a system to classify drugs called Anatomical Therapeutic Chemical (ATC) Classification System (fDSM). This classification system divides drugs into different groups according to the organ or system on which they act and/or to their therapeutic and chemical characteristics. It consists of five hierarchical levels: *anatomical group*, *therapeutic group*, *pharmacological subgroup*, *chemical group*, and *active principle*. So, for instance, the active principle *Enalapril* (ATC code C09AA02) belongs to the chemical group *ACE inhibitors, plain* (C09AA), which is in the pharmacological subgroup C09A with the same name. This subgroup is in the therapeutic group *Agents acting on the renin-angiotensin system* (C09) which belongs to the anatomical group *Cardiovascular system* (C). We have made three modifications to the ATC hierarchy in order to use it as background knowledge:

1. We have added a new level below *active principle* that contains concrete action terms. So for example, *Enalapril Merck 20mg 80 tablets EFG* is an action term that it is a successor of the active principle *Enalapril*.

2. Some drugs are compound having more than one active principle. For example, *Eneas 10/20mg 30 tablets* contains 10mg of *Enalapril* and 20mg of *Nitrendipine*. In the ATC system, these drugs are located in separate groups. However, in our hierarchy they have been introduced as successors of all their active principles. Therefore, *Eneas 10/20mg 30 tablets* is successor of both active principles *Enalapril* and *Nitrendipine*.

3. In order to compare prescriptions of different drugs it is essential to know what is the minimum dose of their respective active principle [1], so we have included this information in the hierarchy.

The resulting extended concept hierarchy is called *pharmacological actions hierarchy*.

Definition 3.1.9 (Pharmacological actions hierarchy) *The pharmacological actions hierarchy is defined as a regular concept hierarchy* $\mathcal{H}_{pA} = (pC \cup pA, \leq, min)$ *(see section 2.5.1.3) where* pC *is the set of concepts in the ATC hierarchy (except the concepts involving compound drugs) and* $pA \subset A$ *is the subset of action terms representing drug prescriptions. The hierarchy reflects with* $\leq$ *the subsumption relations in the ATC hierarchy between the concepts in* pC*. Considering* $pC' \subset pC$ *the concepts representing active principles and given an action term* $a \in pA$*, we have* $c \leq a$ *for each active principle* $c \in pC'$ *that corresponds to the drug prescription* a*. The hierarchy is extended with a function* $min : pC' \rightarrow \mathbb{Q}$ *that matches each active principle with its minimum dose (generally in milligrams).*

Considering the level names of the ATC, the pharmacological actions hierarchy has a schema level order *anatomical group* $<$ *therapeutic group* $<$ *pharmacological subgroup* $<$ *chemical group* $<$ *active principle* $<$ *action term*.

As far as non-pharmacological action terms are concerned, we studied the incorporation of the actions found in the ICD9CM and ICPC systems, but these actions were not specific enough to represent the actions terms found in the databases used in this work. So we created a hierarchical classification for this kind of action terms with the *non-pharmacological actions hierarchy*.

Definition 3.1.10 (Non-pharmacological actions hierarchy) *The non-pharmacological actions hierarchy is defined as a not necessarily regular concept hierarchy* $\mathcal{H}_{nA} = (nC \cup nA, \leq)$ *(see section 2.5.1.3) where* nC *is a set of non-pharmacological concepts and* $nA \subset A$ *is the subset of action terms not representing drug prescriptions. The hierarchy reflects with* $\leq$ *the subsumption relations between the concepts in* nC *and* nA*. The actions terms are located in the lowest levels of the hierarchy (i.e.,* $\forall a \in nA, \nexists c \in nC$ $— a \leq c$*).*

Both hierarchies of pharmacological and non-pharmacological concepts and terms are unified in the *action term hierarchy*.

[1] The minimum dose for each active principle is published in CPGs (SAG02; SAG03)

Definition 3.1.11 (Action term hierarchy) *The action term hierarchy is defined as a concept hierarchy* $\mathcal{H}_A = (C \cup A, \leq, min)$ *(see section 2.5.1.3) where* $C = pC \cup nC \cup \{Action, Pharmacological, Non - pharmacological\}$ *and* $A = pA \cup nA$ *(see definitions 3.1.9 and 3.1.10). This hierarchy contains the sum of all the relationships* $\leq$ *in* $\mathcal{H}_{pA}$ *and* $\mathcal{H}_{nA}$*. Moreover, it has the following relationships:*

- $Action \leq Pharmacological$
- $Action \leq Non - pharmacological$
- $\forall c \in pC$ *in level 1 of* $\mathcal{H}_{pA}$ $Pharmacological \leq c$
- $\forall c \in nC$ *in level 1 of* $\mathcal{H}_{nA}$ $Non - pharmacological \leq c$

The hierarchy also contains the function $min : pC' \rightarrow \mathbb{Q}$ *described in definition 3.1.9.*

Table 3.5 contains a part of a pharmacological actions hierarchy used for the treatment of hypertension, where the elements in the lower level (in italics) are action terms in pA (e.g., *APROVEL 150MG 28 TABLETS*) and the rest of elements are concepts in cA (e.g., C CARDIOVASCULAR SYSTEM). The hierarchy also contains the *min* function defined over the set of active principles (column on the right). Notice that some action terms have a '*' mark meaning that they are compound drugs. These drugs appear more than once in the hierarchy. Concretely they are below each one of their active principles. For example, *COZAAR PLUS 50/12.5 28 COATED TABLETS* is a C09CA01 losartan and a C03AA03 hydrochlorothiazide.

The action term hierarchy presented in the current section establishes some semantic relationships between the different pharmacological and non-pharmacological terms which are necessary to determine the similarity between different treatments. In the next sections we explain how this concept hierarchy can be used to calculate the similarity between two action terms and between two SDA actions, respectively.

3.1.5.1 Calculating the similarity between action terms

The *similarity* $s(a_x, a_y)$ *between two action terms* $a_x, a_y \in A$ is calculated depending on their position in the action term hierarchy $\mathcal{H}_A$. There are five cases:

Case 1: If we compare two action terms a_x and a_y that are exactly the same ($a_x = a_y$), then their similarity is 1 (i.e., $s(a_x, a_y) = 1$).

Table 3.5: Pharmacological actions hierarchy for HT (a part of)

	min (mg)
C CARDIOVASCULAR SYSTEM	
C03 DIURETICS	
C03A LOW-CEILING DIURETICS, THIAZIDES	
C03AA Thiazides, plain	
C03AA03 hydrochlorothiazide	12.5
*COZAAR PLUS 50/12.5 28 COATED TABLETS **	
*CO-DIOVAN 80MG/12.5MG 28 FILM COATED TABLETS **	
*PARAPRES PLUS 16/12.5MG 28 TABLETS **	
*MICARDIS PLUS 80MG/25MG 28 TABLETS **	
*COAPROVEL 300/25MG 28 COATED TABLETS **	
*IXIA PLUS 20/12.5MG 28 FILM COATED TABLETS **	
HIDROSALURETIL 50MG 20 TABLETS	
C09 AGENTS ACTING ON THE RENIN-ANGIOTENSIN SYSTEM	
C09C ANGIOTENSIN II ANTAGONISTS, PLAIN	
C09CA Angiotensin II antagonists, plain	
C09CA01 losartan	25
*COZAAR PLUS 50/12.5 28 COATED TABLETS **	
COZAAR 50MG 28 COATED TABLETS	
COZAAR 12.5MG 7 FILM COATED TABLETS	
C09CA02 eprosartan	600
TEVETENS 600MG 28 COATED TABLETS	
C09CA03 valsartan	80
*CO-DIOVAN 80MG/12.5MG 28 FILM COATED TABLETS **	
DIOVAN 160MG 28 COATED TABLETS	
C09CA04 irbesartan	75
*COAPROVEL 300/25MG 28 COATED TABLETS **	
APROVEL 150MG 28 TABLETS	
C09CA06 candesartan	8
*PARAPRES PLUS 16/12.5MG 28 TABLETS **	
ATACAND 16MG 28 TABLETS	
C09CA07 telmisartan	20
*MICARDIS PLUS 80MG/25MG 28 TABLETS **	
C09CA08 olmesartan medoxomil	10
*IXIA PLUS 20/12.5MG 28 FILM COATED TABLETS **	
IXIA 40MG 28 COATED TABLETS	

* Compound drugs

Case 2: For pharmacological action terms, if we compare the prescription of two drugs a_x and a_y that share the same chemical group, their similarity may be affected by their difference of dose. A first analysis with a dychotomic search using the database available showed that the proportion of similarity that depends on the doses is between 0 and 0.3, being 0 when a_x and a_y show extreme dose differences ($s(a_x, a_y) = 0.7$) and 0.3 when the drugs have equivalent doses ($s(a_x, a_y) = 1$). The similarity between doses of two prescribed drugs is measured considering the minimum doses of their respective active principles. For each active principle a in a chemical group, $\mathcal{H}_A$ contains a minimum dose value which defines the unitary dose (ud) of all the drugs with a. This is calculated with the function min. For example, being a the active principle *Candesartan* its minimum dose is $min(a) = 8$ mg, which is the unitary dose of this active principle. A prescription of *ATACAND 16MG 28 TABLETS* has 16mg per tablet, which is twice the minimum dose, therefore this dose is equal to 2 ud. Sometimes there are prescriptions with doses lower than the minimum dose (e.g., *COZAAR 12.5MG 7 FILM COATED TABLETS* represents 0.5 ud of Losartan) which are usually related to initial treatments. A unitary dose of two drugs of different active principle but of the same chemical group represents an identical pharmacological treatment. See for example in figure 3.1 the equivalences of doses between the active principles *Losartan* and *Valsartan* of the same chemical group *Angiotensin II antagonists, plain* with the respective minimum doses of 25 mg and 80 mg. Notice that their minimum doses represent their unitary dose (1 ud) and determine the equivalence relationship between both active principles.

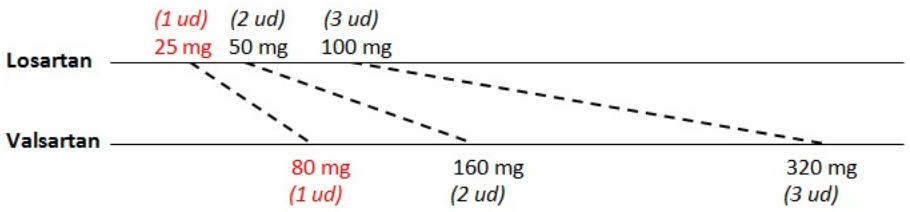


Figure 3.1: Equivalence relationship of dosage between Losartan and Valsartan for some example values

Therefore we can compare doses of drugs of different active principles in a same chemical group as long as they are measured in *ud*'s, and calculate a value of similarity between doses $s_{dose}(a_x, a_y)$. We first express the doses of a_x and a_y in the *ud*'s of their respective active principles (i.e., $a'_x = dose(a_x)/min(active - principle(a_x))$ where $dose(a_x)$ is the dose in the action term a_x and $active - principle(a_x)$ is the active principle of a_x, and equivalently for a'_y), and then we calculate the similarity between doses with equation 3.1.

$$s_{dose}(a_x, a_y) = e^{-\|a'_x - a'_y\|} \tag{3.1}$$

The reduction of similarity between a_x and a_y caused by the difference of doses is equal to $0.3 \cdot (1 - s_{dose}(a_x, a_y))$, so the similarity between two drugs a_x and a_y of the same chemical group is $s(a_x, a_y) = 1 - 0.3 \cdot (1 - s_{dose}(a_x, a_y)) = 0.7 + 0.3 s_{dose}(a_x, a_y)$. In figure 3.2 we can observe the variation of $s_{dose}(a_x, a_y)$ and $s(a_x, a_y)$ for differences of dose lower than 4. Notice that, as we stated before, the similarity between two pharmacological action terms of the same chemical group is always greater than 0.7.

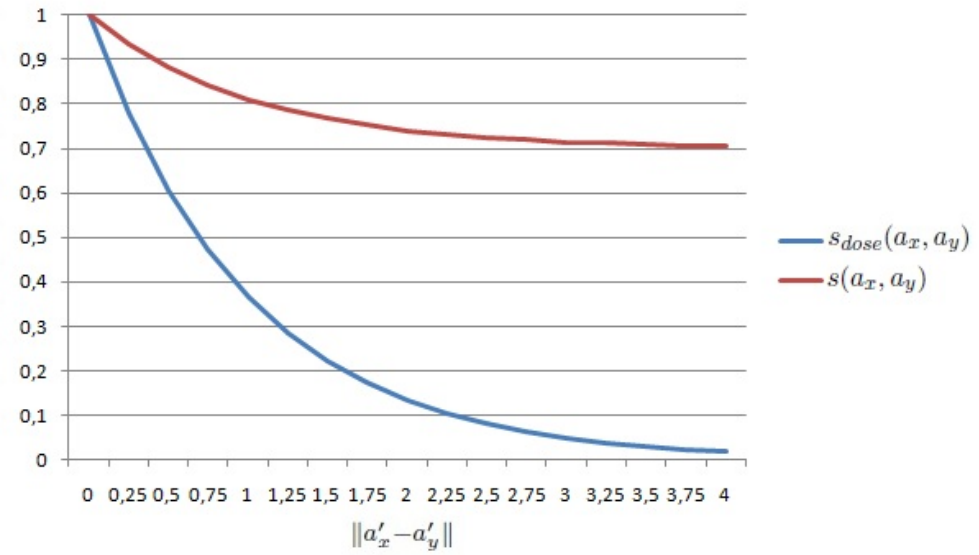


Figure 3.2: Variation of $s_{dose}(a_x, a_y)$ and $s(a_x, a_y)$ when increasing the difference of doses between a_x and a_y

For example, let a_x be *COZAAR 12.5MG 7 FILM COATED TABLETS* and a_y be *DIOVAN 160MG 28 COATED TABLETS* which correspond to drugs with active principles *Losartan* and *Valsartan* respectively. Then a_x has a dose of 0.5 *ud* because 12.5 mg/25 mg is 0.5. Similarly, a_y contains a dose of 2 *ud*. In this case, $s_{dose}(a_x, a_y) = e^{-\|0.5-2\|} = e^{1.5} = 0.22$ and so the similarity between both action terms is $s(a_x, a_y) = 0.77$.

Case 3: Pharmacological action terms with different chemical groups but equal pharmacological subgroup are comparable. Two drugs a_x and a_y in a same pharmacological subgroup are used to treat the same concrete symptoms and so they can be considered partially similar. An analysis of cases of this kind with our data concluded a constant similarity value of 0.5 (i.e., $s(a_x, a_y) = 0.5$).

Case 4: Pharmacological action terms with different pharmacological subgroup but equal therapeutic group may be comparable depending on each concrete case. According to the ATC classification some therapeutic groups cover drugs with similar properties and others which are completely different. We analyzed the therapeutic groups involved in the treatments of hypertension and diabetes mellitus and we concluded the similarity values in table 3.6 for each therapeutic group when two drugs belong to different pharmacological subgroups.

Table 3.6: Similarity values for the drugs in each therapeutic group of the treatments of hypertension and diabetes mellitus

Therapeutic group	**Similarity value**
C02 ANTIHYPERTENSIVES	0.0
C03 DIURETICS	0.3
C07 BETA BLOCKING AGENTS	0.3
C08 CALCIUM CHANNEL BLOCKERS	0.0
C09 AGENTS ACTING ON THE RENIN-ANGIOTENSIN SYSTEM	0.5
C02 ANTIHYPERTENSIVES	0
A10 DRUGS USED IN DIABETES	0

Case 5: Finally, any two action terms a_x and a_y that do not satisfy any of the previous cases are not medically comparable and therefore, their similarity is 0 (i.e., $s(a_x, a_y) = 0$).

3.1.5.2 Calculating the similarity between SDA actions

The *similarity between two SDA actions* $A_x = \{a_{x1}, a_{x2}, ..., a_{xm}\}$ and $A_y = \{a_{y1}, a_{y2}, ..., a_{yn}\}$ according to the action term hierarchy $\mathcal{H}_A$ depends on the similarity between their respective action terms and is calculated with the function $s(A_x, A_y)$. If $s(A_x, A_y) = 0$, A_x and A_y are completely different actions, and if $s(A_x, A_y) = 1$, A_x and A_y are medically equivalent and they can be used interchangeably. To calculate the value of this function we follow three steps:

Step 1: Expanding compound drugs

Step 2: Pairing action terms

Step 3: Calculating the similarity between actions

The first step consists in replacing all the action terms representing prescriptions of compound drugs in the compared actions by action terms with prescriptions of all the drugs that are present in the compound drug, with their corresponding doses. For example, if *MICARDIS PLUS 80MG/25MG 28 TABLETS* is found in one of the actions that are being compared, this action term is replaced by *TELMISARTAN 80MG 28 TABLETS* and *HYDROCHLOROTHIAZIDE 25MG 28 TABLETS*.

Once all the prescriptions of compound drugs in A_x and A_y have been replaced by the prescriptions of their corresponding single drugs, the action terms in A_x are paired with the action terms in A_y. The aim is to find semantically equivalent action terms from both interventions. Formally, we want to create a set P of pairs (a_p, a_q) with $a_p \in A_x$ and $a_q \in A_y$ such that (1) $\forall a \in A_x, \exists!(a, a_q) \in P$, (2) $\forall a \in A_y, \exists!(a_p, a) \in P$, and (3) $s(a_p, a_q) > 0$. All the action terms in a SDA action are relevant an so, in order to reach a successful pairing, A_x and A_y must contain the same number of action terms ($m = n$). At this point, if two actions A_x and A_y have a different number of action terms we can conclude that $s(A_x, A_y) = 0$. Suppose that A_x and A_y have both n action terms, then the pairing is performed as follows. For the first action term a_{x1} in A_x we calculate its similarity to each one of the action terms in A_y. If $\forall a_{yi} \in A_y, s(a_{x1}, a_{yi}) = 0$ we cannot pair a_{x1} with an equivalent action term in A_y, therefore we conclude that $s(A_x, A_y) = 0$. Otherwise, we create a pair (a_{x1}, a_{yj}) where a_{yj} is the most similar action term to a_{x1} in A_y (i.e., $a_{yj} = \arg\max_{a_{yi} \in A_y} s(a_{x1}, a_{yi})$). Then, after discarding

the actions that have already been paired, we repeat this procedure until n pairs are created.

If the pairing has succeeded then $s(A_x, A_y) > 0$. The final value of similarity is calculated as the average of similarities between the action terms in each pair in P with equation 3.2.

$$s(A_x, A_y) = \frac{1}{n} \sum_{(a_p, a_q) \in P} s(a_p, a_q) \tag{3.2}$$

We use this similarity function to determine whether two SDA actions are equivalent or not by specifying a similarity threshold δ between 0 and 1. If $s(A_x, A_y) \geq \delta$ then A_x and A_y are considered equivalent SDA actions.

Figure 3.3 depicts an example of applying the previous procedure to determine the similarity between two actions A_x and A_y. These actions are completely equivalent because they both contain a compound drug with 2 *ud* of hydrochlorothiazide (25mg) and 4 *ud* of active principle of Angiotensin II antagonists, plain (300mg and 80mg of Irbesartan and Telmisartan, respectively), and also the non-pharmacological action *Education*.

In figure 3.4 there is a less obvious example of completely equivalent actions. These actions contain a different compound drug and a different single drug. Once expanded we observe that they actually have the same active principles and dosages.

The calculation of similarities between actions has lots of applications to solve other medical problems out of the scope of this thesis. One of these applications is the reduction of treatment costs by detecting dominant alternatives which is presented in (LVRC12b).

3.1.5.3 Calculating the homogeneity of a set of treatments

The similarity between actions can be applied to calculate the *homogeneity of the treatments* within a multiset of actions $A' = \{A_1, A_2, ...\}$ of length n. With the previous similarity function we calculate similarities between each pair of actions. The homogeneity h of the treatments in A' according to the action term hierarchy $\mathcal{H}_A$ can be defined as in equation 3.3.

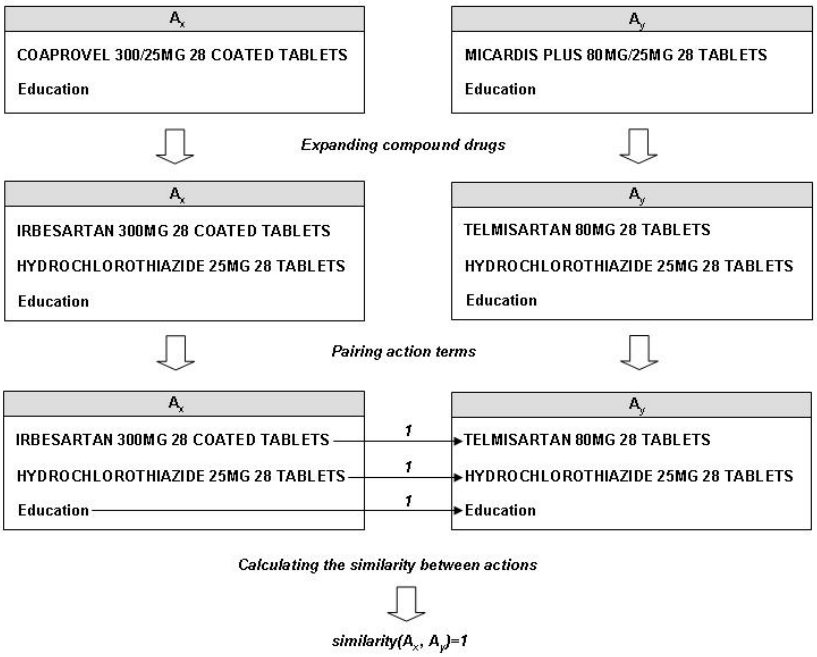


Figure 3.3: Example 1 of determining the similarity between two SDA actions

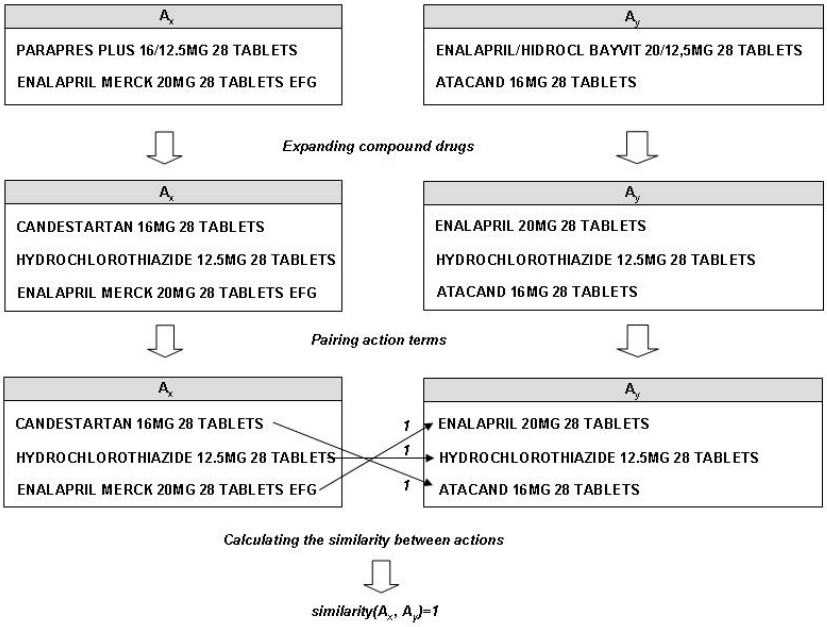


Figure 3.4: Example 2 of determining the similarity between two SDA actions

$$h(A') = \frac{1}{n^2} \sum_{A_i \in A'} \sum_{A_j \in A'} s(A_i, A_j) \tag{3.3}$$

We can also calculate the homogeneity with the alternative equation 3.4. In this equation the homogeneity of the set of treatments is equal to the lowest similarity between two of them. This guarantees a minimum similarity between all the treatments, but here we will not use it because it is too much restrictive for the purpose of this thesis.

$$h(A') = \min_{A_i, A_j \in A'} s(A_i, A_j) \tag{3.4}$$

3.2 Summary of background knowledge

This chapter has introduced the medical background knowledge needed to support the automatic generation of SDA diagrams. We have presented the different kinds of background knowledge needed, as well as their formalization as knowledge structures. A summary of all the background knowledge required is shown in the following list.

a) Background knowledge related to states:

- For the constraints on health care states: a state constraints graph G_S containing state term constraints.
- For preference between state terms: a state terms partial order $\leq_S$.

b) Background knowledge related to decisions:

- For semantic decisions: a semantic decisions hypergraph H_D containing the possible semantic decisions.
- For order of decision sequences: a decisions partial order $\leq_D$.

c) Background knowledge related to actions:

- For similarity between actions: an action terms hierarchy $\mathcal{H}_A$ that is used to calculate the values of the similarity function s and the homogeneity function h.

The formalization of this background knowledge has been included in the paper (LVRC12a).

3.3 Background knowledge formalization for hypertension, diabetes mellitus and their comorbidity

This section contains the structures created for the diseases of hypertension, diabetes mellitus, and the comorbidity of both diseases to represent the required background knowledge introduced in this chapter: the state constraints graph G_S with the constraints on state terms, the state terms partial order $\leq_S$ that specifies the preference between state terms, the semantic decisions hypergraph H_D, the decisions partial order $\leq_D$ over the different semantic decisions and the action terms hierarchy $\mathcal{H}_A$ which will be used to calculate the values of the similarity function s and the homogeneity function h.

This knowledge repository has been created together with health care professionals from the SAGESSA Health-Care Group (SAG) using their own preferences and experience, and also the evidence-based knowledge contained in other resources like CPGs (SAG02; SAG03) or the Anatomical Therapeutic Chemical (ATC) Classification System (fDSM).

3.3.1 Hypertension

3.3.1.1 Constraints on health care states

Table 3.7 contains the state constraints graph for hypertension. In order to reduce the size of the table, the terms that have no constraints have been replaced by '...'. The patient is considered to have heart risk depending on his age, sex and the presence of LVH. Other risk factors related to smoking habits or hypercholesterolemia have been ignored because here we only consider pure hypertensive patients with no other pathologies or complications. Therefore, if we use heart risk to describe the states of the SDA diagram we do not want to use the different signs that may imply high risk because it could cause redundancy within a state and different levels of abstraction in the terminology used by the whole set of states.

3.3.1.2 Preference between state terms

The state terms partial order for hypertension is shown in table 3.8. The health care professionals decided to give more priority to the state terms regarding the situation

Table 3.7: State constraints graph for HT

	HEART RISK	NOT HEART RISK	< 40 YEARS OLD	40 .. 65 YEARS OLD	65 .. 75 YEARS OLD	> 75 YEARS OLD	FEMALE	MALE	LVH	NO LVH	...
H. RISK			X	X	X	X	X	X	X	X	
NOT H. RISK			X	X	X	X	X	X	X	X	
< 40 Y. OLD											
40 .. 65 Y. OLD											
65 .. 75 Y. OLD											
> 75 Y. OLD											
FEMALE											
MALE											
LVH											
NO LVH											
...											

of the patient within the treatment of hypertension (e.g., FOLLOWING HEALTHY HABITS, TAKING 2 DRUGS) rather than to the control of the disease (e.g., NOT CONTROLLED DBP, CONTROLLED SBP).

Table 3.8: State terms partial order for HT

Priority	State term
1	NOT FOLLOWING HEALTHY HABITS
	FOLLOWING HEALTHY HABITS
	NOT TAKING MEDICATION
	TAKING 1 DRUG
	TAKING 2 DRUGS
	TAKING 3 DRUGS
2	CONTROLLED DBP
	NOT CONTROLLED DBP
	CONTROLLED SBP
	NOT CONTROLLED SBP
	HEART RISK
	NOT HEART RISK
3	< 40 YEARS OLD
	40 .. 65 YEARS OLD
	65 .. 75 YEARS OLD
	> 75 YEARS OLD
	LVH
	NO LVH
	MALE
	FEMALE

3.3.1.3 Semantic decisions

Table 3.9 contains the semantic decisions hypergraph for hypertension. Basically, the different decision terms have been grouped in semantic decisions according to the evidence they are related to.

Table 3.9: Semantic decisions hypergraph for HT

SD name	SD terms
Abdominal exploration	NORMAL ABDOMINAL EXPLORATION
	NOT NORMAL ABDOMINAL EXPLORATION
Adverse effects	ADVERSE EFFECTS
	NOT ADVERSE EFFECTS
Age	< 40 YEARS OLD
	40 .. 65 YEARS OLD
	65 .. 75 YEARS OLD
	> 75 YEARS OLD
Alcohol	ALCOHOL
	NOT ALCOHOL
BMI	NORMAL BMI
	OVERWEIGHT BMI
	OBESE BMI
Cardiac auscultation	NORMAL CARDIAC AUSCULTATION
	NOT NORMAL CARDIAC AUSCULTATION
Correct medication	TAKING MEDICATION CORRECTLY
	NOT TAKING MEDICATION CORRECTLY
DBP	CONTROLLED DBP
	NOT CONTROLLED DBP
ECG	ALTERED ECG
	NORMAL ECG
Glucose	LOW GLUCOSE
	NORMAL GLUCOSE
	HIGH GLUCOSE
GOT	NORMAL GOT
	HIGH GOT
GPT	LOW GPT
	NORMAL GPT
	HIGH GPT
Healthy habits	FOLLOWING HEALTHY HABITS
	NOT FOLLOWING HEALTHY HABITS
Heart rate	LOW HEART RATE
	NORMAL HEART RATE
	HIGH HEART RATE
Heart risk	HEART RISK
	NOT HEART RISK
LVH	LVH
	NO LVH

Continued on next page

Table 3.9 – continued from previous page

SD name	SD terms
Medication	NOT TAKING MEDICATION
	TAKING 1 DRUG
	TAKING 2 DRUGS
	TAKING 3 DRUGS
Physical activity	ADEQUATE PHYSICAL ACTIVITY
	INADEQUATE PHYSICAL ACTIVITY
Proteinuria	NORMAL PROTEINURIA
	HIGH PROTEINURIA
Pulmonary auscultation	NORMAL PULMONARY AUSCULTATION
	NOT NORMAL PULMONARY AUSCULTATION
SBP	CONTROLLED SBP
	NOT CONTROLLED SBP
Sex	MALE
	FEMALE
Tibial oscillometry	LOW TIBIAL OSCILLOMETRY
	HIGH TIBIAL OSCILLOMETRY

3.3.1.4 Order of decision sequences

Table 3.10 shows the decisions partial order for hypertension. The semantic decisions of priorities 1 and 2 are used to decide on what kind of treatment is needed. The third level of priority contains the semantic decision Heart rate which can be used to discard certain types of drugs, and finally, the rest of terms are used to refine the treatment.

Table 3.10: Decisions partial order for HT

Priority	SD
1	DBP
	Healthy habits
	Medication
	SBP
2	Adverse effects
	Age
	Correct medication
	Heart risk
	Physical activity
	Sex
3	Heart rate
4	Abdominal exploration
	Alcohol
	BMI
	Cardiac auscultation
	ECG

Continued on next page

Table 3.10 – continued from previous page

Priority	SD
	Glucose
	GOT
	GPT
	LVH
	Proteinuria
	Pulmonary auscultation
	Tibial oscillometry

3.3.1.5 Similarity between actions

The action hierarchy that contains the pharmacological and non-pharmacological actions for the treatment of hypertension is shown in table 3.11.

Table 3.11: Action hierarchy for HT

	min (mg)
Action	
Pharmacological	
C CARDIOVASCULAR SYSTEM	
C02 ANTIHYPERTENSIVES	
C02C ANTIADRENERGIC AGENTS, PERIPHERALLY ACTING	
C02CA Alpha-adrenoreceptor antagonists	
C02CA04 doxazosin	1
CARDURAN 4MG 28 TABLETS	
...	
C03 DIURETICS	
C03A LOW-CEILING DIURETICS, THIAZIDES	
C03AA Thiazides, plain	
C03AA03 hydrochlorothiazide	12.5
*COZAAR PLUS 50/12.5 28 COATED TABLETS **	
*CO-DIOVAN 80MG/12.5MG 28 FILM COATED TABLETS **	
*PARAPRES PLUS 16/12.5MG 28 TABLETS **	
*MICARDIS PLUS 80MG/25MG 28 TABLETS **	
*COAPROVEL 300/25MG 28 COATED TABLETS **	
*IXIA PLUS 20/12.5MG 28 FILM COATED TABLETS **	
*AMERIDE 5/50MG 60 TABLETS **	
*ENALAPRIL/HIDROCL BAYVIT 20/12.5MG 28 TABLETS **	
*ZESTORETIC 20/12.5MG 28 TABLETS **	
HIDROSALURETIL 50MG 20 TABLETS	
*EMCORETIC 10 MG/25 MG 56 COATED TABLETS **	
...	
C03B LOW-CEILING DIURETICS, EXCL. THIAZIDES	
C03BA Sulfonamides, plain	
C03BA04 chlortalidone	12.5

Continued on next page

Table 3.11 – continued from previous page

	min (mg)
*BLOKIUM-DIU 28 TABLETS **	
HIGROTONA 50MG 30 TABLETS	
...	
C03BA11 indapamide	1.25
TERTENSIF 2.5MG 30 COATED TABLETS	
...	
C03C HIGH-CEILING DIURETICS	
C03CA Sulfonamides, plain	
C03CA01 furosemide	40
FUROSEMIDA CINFA 40MG 30 TABLETS EFG	
...	
C03CA04 torasemide	2.5
SUTRIL 10MG 30 TABLETS	
...	
C03D POTASSIUM-SPARING AGENTS	
C03DA Aldosterone antagonists	
C03DA01 spironolactone	25
ALDACTONE 25 MG 20 FILM COATED TABLETS	
...	
C03DB Other potassium-sparing agents	
C03DB01 amiloride	2.5
*AMERIDE 5/50MG 60 TABLETS **	
...	
C07 BETA BLOCKING AGENTS	
C07A BETA BLOCKING AGENTS	
C07AA Beta blocking agents, non-selective	
C07AA05 propranolol	40
SUMIAL 10MG 50 TABLETS	
...	
C07AB Beta blocking agents, selective	
C07AB02 metoprolol	50
BELOKEN 100MG 40 TABLETS	
*LOGIMAX 5/50MG 30 TABLETS **	
...	
C07AB03 atenolol	25
*BLOKIUM-DIU 28 TABLETS **	
ATENOLOL ALTER 50MG 60 TABLETS EFG	
...	
C07AB07 bisoprolol	2.5
EMCONCOR 5MG 30 COATED TABLETS	
*EMCORETIC 10 MG/25 MG 56 COATED TABLETS **	
...	
C07AB12 nebivolol	2.5
LOBIVON 5MG 28 TABLETS	
...	
C07AG Alpha and beta blocking agents	
C07AG02 carvedilol	12.5
COROPRES 25MG 28 TABLETS	
...	

Continued on next page

Table 3.11 – continued from previous page

	min (mg)
C08 CALCIUM CHANNEL BLOCKERS	
C08C SELECTIVE CALCIUM CHANNEL BLOCKERS WITH MAINLY VASCULAR EFFECTS	
C08CA Dihydropyridine derivatives	
C08CA01 amlodipine	2.5
ASTUDAL 10MG 30 TABLETS	
*EXFORGE 5MG/160MG 28 FILM COATED TABLETS **	
...	
C08CA02 felodipine	2.5
PLENDIL 5MG 30 TABLETS	
*LOGIMAX 5/50MG 30 TABLETS **	
...	
C08CA05 nifedipine	30
ADALAT OROS 30MG 28 TABLETS	
...	
C08CA08 nitrendipine	10
BAYPRESOL 20MG 30 TABLETS	
...	
C08CA12 barnidipine	10
BARNIX 20MG 56 CAPSULES	
...	
C08D SELECTIVE CALCIUM CHANNEL BLOCKERS WITH DIRECT CARDIAC EFFECTS	
C08DA Phenylalkylamine derivatives	
C08DA01 verapamil	120
MANIDON 80MG 60 FILM-COATED TABLETS	
...	
C08DB Benzothiazepine derivatives	
C08DB01 diltiazem	120
UNI MASDIL 200 MG 28 CAPSULES	
...	
C09 AGENTS ACTING ON THE RENIN-ANGIOTENSIN SYSTEM	
C09A ACE INHIBITORS, PLAIN	
C09AA ACE inhibitors, plain	
C09AA01 captopril	25
CAPTOPRIL STADA 25MG 60 TABLETS EFG	
...	
C09AA02 enalapril	5
ENALAPRIL MERCK 20MG 28 TABLETS	
*ENALAPRIL/HIDROCL BAYVIT 20/12.5MG 28 TABLETS **	
...	
C09AA03 lisinopril	5
LISINOPRIL MYLAN 20 MG 28 TABLETS	
*ZESTORETIC 20/12.5MG 28 TABLETS **	
...	
C09AA04 perindopril	2
COVERSYL 4MG 30 TABLETS	
...	
C09AA05 ramipril	1.25
ACOVIL 5MG 28 TABLETS	
...	

Continued on next page

Table 3.11 – continued from previous page

	min (mg)
C09AA06 quinapril	5
ECTREN 20MG 28 COATED TABLETS	
...	
C09AA16 imidapril	2.5
HIPERTENE 10MG 28 TABLETS	
...	
C09C ANGIOTENSIN II ANTAGONISTS, PLAIN	
C09CA Angiotensin II antagonists, plain	
C09CA01 losartan	25
*COZAAR PLUS 50/12.5 28 COATED TABLETS **	
COZAAR 50MG 28 COATED TABLETS	
...	
C09CA02 eprosartan	600
TEVETENS 600MG 28 COATED TABLETS	
...	
C09CA03 valsartan	80
*CO-DIOVAN 80MG/12.5MG 28 FILM COATED TABLETS **	
*EXFORGE 5MG/160MG 28 FILM COATED TABLETS **	
DIOVAN 160MG 28 COATED TABLETS	
...	
C09CA04 irbesartan	75
*COAPROVEL 300/25MG 28 COATED TABLETS **	
APROVEL 150MG 28 TABLETS	
...	
C09CA06 candesartan	8
*PARAPRES PLUS 16/12.5MG 28 TABLETS **	
ATACAND 16MG 28 TABLETS	
...	
C09CA07 telmisartan	20
*MICARDIS PLUS 80MG/25MG 28 TABLETS **	
...	
C09CA08 olmesartan medoxomil	10
*IXIA PLUS 20/12.5MG 28 FILM COATED TABLETS **	
IXIA 40MG 28 COATED TABLETS	
...	
C09X OTHER AGENTS ACTING ON THE RENIN-ANGIOTENSIN SYSTEM	
C09XA Renin-inhibitors	
C09XA02 aliskiren	150
RASILEZ 150MG 28 COATED TABLETS	
...	
Non-pharmacological	
Education	
Verification	
Check	
Follow-up	
Evaluation of risk factors	
Consultation	
Consultation cardiology	
Complementary tests	

Continued on next page

Table 3.11 – continued from previous page

	min (mg)
Laboratory	
ECG	
Radiology	
Other procedures	
Tibial post. D(X)	

* Compound drugs

3.3.2 Diabetes mellitus

3.3.2.1 Constraints on health care states

No constraints on health care states have been required by health care professionals in the case of diabetes mellitus.

3.3.2.2 Preference between state terms

The state terms partial order for diabetes is shown in table 3.12. In this case, both the situation of the patient within the treatment (e.g., FOLLOWING HEALTHY HABITS, TAKING INSULINS) and the control of the disease in terms of glucose (NORMAL/HIGH GLUCOSE) are of major interest for health care professionals and so, they are given priority 1. It would be more medically logical to consider the level of HbAlC before the level of glucose as an indicator of the control of the disease, but the health care professionals argued that, due to the low rate of encounters in their databases that contain the level of HbAlC, it would be more suitable to use the level of glucose. This is a clear example of using background knowledge not only to represent theorical medical knowledge but also to express concrete preferences of a certain health care center.

3.3.2.3 Semantic decisions

Table 3.13 contains the semantic decisions hypergraph for diabetes where the different decision terms have been grouped in semantic decisions according to the evidence they are related to.

Table 3.12: State terms partial order for DM

Priority	State term
1	NOT FOLLOWING HEALTHY HABITS
	FOLLOWING HEALTHY HABITS
	NOT TAKING MEDICATION
	TAKING OHDS
	TAKING INSULINS
	TAKING OHDS+INSULINS
	NORMAL GLUCOSE
	HIGH GLUCOSE
2	NORMAL HBALC
	HIGH HBALC

Table 3.13: Semantic decisions hypergraph for DM

SD name	SD terms
Abdominal exploration	NORMAL ABDOMINAL EXPLORATION
	NOT NORMAL ABDOMINAL EXPLORATION
Adverse effects	ADVERSE EFFECTS
	NOT ADVERSE EFFECTS
Age	$<$ 40 YEARS OLD
	40 .. 65 YEARS OLD
	65 .. 75 YEARS OLD
	$>$ 75 YEARS OLD
BMI	NORMAL BMI
	OVERWEIGHT BMI
	OBESE BMI
Cardiac auscultation	NORMAL CARDIAC AUSCULTATION
	NOT NORMAL CARDIAC AUSCULTATION
Correct medication	TAKING MEDICATION CORRECTLY
	NOT TAKING MEDICATION CORRECTLY
Diet	FOLLOWING DIET
	NOT FOLLOWING DIET
ECG	ALTERED ECG
	NORMAL ECG
Foot	ALTERED FOOT EXPLORATION
	NORMAL FOOT EXPLORATION
Glucose	NORMAL GLUCOSE
	HIGH GLUCOSE
HbAlC	NORMAL HBALC
	HIGH HBALC
Healthy habits	FOLLOWING HEALTHY HABITS
	NOT FOLLOWING HEALTHY HABITS
Heart rate	LOW HEART RATE
	NORMAL HEART RATE
	HIGH HEART RATE
Heart risk	HEART RISK
	NOT HEART RISK

Continued on next page

Table 3.13 – continued from previous page

SD name	SD terms
LVH	LVH
	NO LVH
Medication	NOT TAKING MEDICATION
	TAKING OHDS
	TAKING INSULINS
	TAKING OHDS+INSULINS
Nutrition	NUTRITIONAL RISK
	NOT NUTRITIONAL RISK
Physical activity	ADEQUATE PHYSICAL ACTIVITY
	INADEQUATE PHYSICAL ACTIVITY
Proteinuria	NORMAL PROTEINURIA
	HIGH PROTEINURIA
Pulmonary auscultation	NORMAL PULMONARY AUSCULTATION
	NOT NORMAL PULMONARY AUSCULTATION
Rhythmic heart	RHYTHMIC HEART
	ARRHYTHMIC HEART
Sex	MALE
	FEMALE

3.3.2.4 Order of decision sequences

Table 3.14 shows the decisions partial order for diabetes. The semantic decisions of priority 1 basically determine the treatment of the patient, while those of priority 2 are used to refine it.

Table 3.14: Decisions partial order for DM

Priority	SD
1	Adverse effects
	Age
	BMI
	Correct medication
	Diet
	Glucose
	HbAlC
	Healthy habits
	Medication
	Nutrition
	Physical activity
	Sex
2	Abdominal exploration
	Cardiac auscultation
	Foot
	Heart rate

Continued on next page

Table 3.14 – continued from previous page

Priority	SD
	Pulmonary auscultation
	Rhythmic heart
	Proteinuria
	ECG
	Heart risk
	LVH

3.3.2.5 Similarity between actions

The action hierarchy that contains the pharmacological and non-pharmacological actions for the treatment of diabetes is shown in table 3.15.

Table 3.15: Action hierarchy for DM

	min (mg*)
Action	
Pharmacological	
A ALIMENTARY TRACT AND METABOLISM	
A10 DRUGS USED IN DIABETES	
A10A INSULINS AND ANALOGUES	
A10AB Insulins and analogues for injection, fast-acting	
A10AB01 insulin (human)	100
ACTRAPID INNOLET 100UI/ML	
...	
A10AB05 insulin aspart	100
NOVORAPID FLEXPEN 100UI/ML	
...	
A10AC Insulins and analogues for injection, intermediate-acting	
A10AC01 insulin (human)	100
INSULATARD NPH FLEXPEN 100UI/ML	
...	
A10AD Insulins and analogues for injection, intermediate-acting combined with fast-acting	
A10AD01 insulin (human)	100
MIXTARD 30 INNOLET 100UI/ML	
...	
A10AD05 insulin aspart	100
NOVOMIX 30 FLEXPEN 100UI/ML	
...	
A10AE Insulins and analogues for injection, long-acting	
A10AE04 insulin glargine	100
LANTUS 100UI/ML OPTISET	
...	
A10B BLOOD GLUCOSE LOWERING DRUGS, EXCL. INSULINS	
A10BA Biguanides	

Continued on next page

Table 3.15 – continued from previous page

	min (mg*)
A10BA02 metformin	850
DIANBEN 850MG 50 TABLETS	
...	
A10BB Sulfonamides, urea derivatives	
A10BB01 glibenclamide	2.5
EUGLUCON 5MG 100 TABLETS	
...	
A10BB09 gliclazide	40
DIAMICRON 80MG 60 TABLETS	
...	
A10BB12 glimepiride	1
AMARYL 4MG 120 TABLETS	
...	
A10BF Alpha glucosidase inhibitors	
A10BF01 Acarbose	75
GLUMIDA 100MG 100 TABLETS	
...	
A10BG Thiazolidinediones	
A10BG02 rosiglitazone	1
AVANDIA 4MG 28 TABLETS	
...	
A10BG03 pioglitazone	15
ACTOS 30MG 56 TABLETS	
...	
A10BH Dipeptidyl peptidase 4 (DPP-4) inhibitors	
A10BH01 sitagliptin	100
JANUVIA 100MG 56 FILM COATED TABLETS	
...	
Non-pharmacological	
Education	
Verification	
Check	
Follow-up	
Evaluation of risk factors	
Consultation	
Consultation endocrinology	
Consultation ophthalmology	
Complementary tests	
Laboratory	
ECG	
Radiology	
Other procedures	
Lower extremity oscillometry	

* UI/ml for A10A INSULINS AND ANALOGUES

3.3.3 Hypertension + Diabetes mellitus

3.3.3.1 Constraints on health care states

The state constraints graph for hypertension plus diabetes mellitus is the same that for hypertension (see table 3.7). No constraints were needed for the case of diabetes mellitus, and no additional constraints were found when combining hypertension with diabetes mellitus.

3.3.3.2 Preference between state terms

Health care professionals proposed two state terms partial orders in order to generate to different SDA diagrams for hypertension plus diabetes mellitus that gave different visions of the treatment. The first one is shown in table 3.16 and it gives more priority to the terms related to the situation of the patient within the treatment of the diseases (e.g., FOLLOWING HEALTHY HABITS, TAKING OHDS+1 HYPOTENSIVE DRUG).

Table 3.16: State terms partial order for HT+DM (1)

Priority	State term
1	NOT FOLLOWING HEALTHY HABITS
	FOLLOWING HEALTHY HABITS
	NOT TAKING MEDICATION
	TAKING OHDS
	TAKING INSULINS
	TAKING OHDS+INSULINS
	TAKING 1 HYPOTENSIVE DRUG
	TAKING 2 HYPOTENSIVE DRUGS
	TAKING OHDS+1 HYPOTENSIVE DRUG
	TAKING OHDS+2 HYPOTENSIVE DRUGS
	TAKING OHDS+INSULINS+2 HYPOTENSIVE DRUGS
2	CONTROLLED BP
	NOT CONTROLLED BP
	NORMAL GLUCOSE
	HIGH GLUCOSE
3	$<$ 40 YEARS OLD
	40 .. 65 YEARS OLD
	65 .. 75 YEARS OLD
	$>$ 75 YEARS OLD
	HEART RISK
	NOT HEART RISK
	LVH
	NO LVH

Continued on next page

Table 3.16 – continued from previous page

Priority	State term
	MALE
	FEMALE
	NORMAL HBALC
	HIGH HBALC

Table 3.17 contains the other state terms partial order which gives more priority to the control of the diseases (e.g., CONTROLLED BP, HIGH GLUCOSE).

Table 3.17: State terms partial order for HT+DM (2)

Priority	State term
1	CONTROLLED BP
	NOT CONTROLLED BP
	NORMAL GLUCOSE
	HIGH GLUCOSE
2	NOT FOLLOWING HEALTHY HABITS
	FOLLOWING HEALTHY HABITS
	NOT TAKING MEDICATION
	TAKING OHDS
	TAKING INSULINS
	TAKING OHDS+INSULINS
	TAKING 1 HYPOTENSIVE DRUG
	TAKING 2 HYPOTENSIVE DRUGS
	TAKING OHDS+1 HYPOTENSIVE DRUG
	TAKING OHDS+2 HYPOTENSIVE DRUGS
	TAKING OHDS+INSULINS+2 HYPOTENSIVE DRUGS
3	$<$ 40 YEARS OLD
	40 .. 65 YEARS OLD
	65 .. 75 YEARS OLD
	$>$ 75 YEARS OLD
	HEART RISK
	NOT HEART RISK
	LVH
	NO LVH
	MALE
	FEMALE
	NORMAL HBALC
	HIGH HBALC

3.3.3.3 Semantic decisions

Table 3.18 contains the semantic decisions hypergraph for hypertension plus diabetes mellitus. Basically, the different decision terms have been grouped in semantic decisions according to the evidence they are related to.

Table 3.18: Semantic decisions hypergraph for HT+DM

SD name	SD terms
Abdominal exploration	NORMAL ABDOMINAL EXPLORATION
	NOT NORMAL ABDOMINAL EXPLORATION
Adverse effects	ADVERSE EFFECTS
	NOT ADVERSE EFFECTS
Age	< 40 YEARS OLD
	40 .. 65 YEARS OLD
	65 .. 75 YEARS OLD
	> 75 YEARS OLD
Alcohol	NOT ALCOHOL
	ALCOHOL
BMI	NORMAL BMI
	OVERWEIGHT BMI
	OBESE BMI
BP	CONTROLLED BP
	NOT CONTROLLED BP
Cardiac auscultation	NORMAL CARDIAC AUSCULTATION
	NOT NORMAL CARDIAC AUSCULTATION
Correct medication	TAKING MEDICATION CORRECTLY
	NOT TAKING MEDICATION CORRECTLY
Diet	FOLLOWING DIET
	NOT FOLLOWING DIET
ECG	ALTERED ECG
	NORMAL ECG
Foot	ALTERED FOOT EXPLORATION
	NORMAL FOOT EXPLORATION
Glucose	NORMAL GLUCOSE
	HIGH GLUCOSE
GOT	NORMAL GOT
	HIGH GOT
GPT	LOW GPT
	NORMAL GPT
	HIGH GPT
HbAlC	NORMAL HBALC
	HIGH HBALC
Healthy habits	NOT FOLLOWING HEALTHY HABITS
	FOLLOWING HEALTHY HABITS
Heart rate	LOW HEART RATE
	NORMAL HEART RATE
	HIGH HEART RATE

Continued on next page

Table 3.18 – continued from previous page

SD name	SD terms
Heart risk	HEART RISK
	NOT HEART RISK
LVH	LVH
	NO LVH
Medication	NOT TAKING MEDICATION
	NOT TAKING MEDICATION
	TAKING OHDS
	TAKING INSULINS
	TAKING OHDS+INSULINS
	TAKING 1 HYPOTENSIVE DRUG
	TAKING 2 HYPOTENSIVE DRUGS
	TAKING OHDS+1 HYPOTENSIVE DRUG
	TAKING OHDS+2 HYPOTENSIVE DRUGS
	TAKING OHDS+INSULINS+2 HYPOTENSIVE DRUGS
Nutrition	NUTRITIONAL RISK
	NOT NUTRITIONAL RISK
Physical activity	ADEQUATE PHYSICAL ACTIVITY
	INADEQUATE PHYSICAL ACTIVITY
Proteinuria	NORMAL PROTEINURIA
	HIGH PROTEINURIA
Pulmonary auscultation	NORMAL PULMONARY AUSCULTATION
	NOT NORMAL PULMONARY AUSCULTATION
Rhythmic heart	RHYTHMIC HEART
	ARRHYTHMIC HEART
Sex	MALE
	FEMALE
Tibial oscillometry	LOW TIBIAL OSCILLOMETRY
	HIGH TIBIAL OSCILLOMETRY

3.3.3.4 Order of decision sequences

Table 3.19 shows the decisions partial order for the treatment of hypertension and diabetes mellitus. The semantic decisions of priority 1 are used to decide whether pharmacological treatment is required or not, or if it must be changed. Those of priority 2 may determine the type of treatment (e.g., a young patient should be treated with insulin, an obese patient should be treated with metformin). Finally, the semantic decisions of priority 3 can be used to decide details as for example the dosage.

Table 3.19: Decisions partial order for HT+DM

Priority	SD
1	BP
	Continued on next page

Table 3.19 – continued from previous page

Priority	SD
	Glucose
	Healthy habits
	Medication
2	Age
	BMI
	Diet
	HbAlC
	Nutrition
	Physical activity
	Sex
3	Abdominal exploration
	Adverse effects
	Alcohol
	Cardiac auscultation
	Correct medication
	ECG
	Foot
	GOT
	GPT
	Heart rate
	Heart risk
	LVH
	Proteinuria
	Pulmonary auscultation
	Rhythmic heart
	Tibial oscillometry

3.3.3.5 Similarity between actions

The action hierarchy that contains the pharmacological and non-pharmacological actions for the treatment of patients with hypertension and diabetes is shown in table 3.20.

Table 3.20: Action hierarchy for HT+DM

	min (mg**)
Action	
Pharmacological	
A ALIMENTARY TRACT AND METABOLISM	
A10 DRUGS USED IN DIABETES	
A10A INSULINS AND ANALOGUES	
A10AB Insulins and analogues for injection, fast-acting	
A10AB05 insulin aspart	100
NOVORAPID FLEXPEN 100UI/ML	
...	

Continued on next page

Table 3.20 – continued from previous page

	min (mg**)
A10AC Insulins and analogues for injection, intermediate-acting	
A10AC01 insulin (human)	100
INSULATARD NPH FLEXPEN 100UI/ML	
...	
A10AD Insulins and analogues for injection, intermediate-acting combined with fast-acting	
A10AD05 insulin aspart	100
NOVOMIX 30 FLEXPEN 100UI/ML	
...	
A10AE Insulins and analogues for injection, long-acting	
A10AE04 insulin glargine	100
LANTUS 100UI/ML OPTISET	
...	
A10AE05 insulin detemir	100
LEVEMIR 100U/ML INNOLET	
...	
A10B BLOOD GLUCOSE LOWERING DRUGS, EXCL. INSULINS	
A10BA Biguanides	
A10BA02 metformin	850
DIANBEN 850MG 50 TABLETS	
*AVANDAMET 2MG/500MG 112 TABLETS **	
*EUCREAS 50MG/850MG 60 TABLETS **	
...	
A10BB Sulfonamides, urea derivatives	
A10BB01 glibenclamide	2.5
EUGLUCON 5MG 100 TABLETS	
...	
A10BB09 gliclazide	40
DIAMICRON 80MG 60 TABLETS	
...	
A10BB12 glimepiride	1
AMARYL 4MG 120 TABLETS	
...	
A10BF Alpha glucosidase inhibitors	
A10BF01 Acarbose	75
GLUMIDA 100MG 100 TABLETS	
...	
A10BF02 Miglitol	75
DIASTABOL 100MG 90 TABLETS	
...	
A10BG Thiazolidinediones	
A10BG02 rosiglitazone	1
AVANDIA 4MG 28 TABLETS	
*AVANDAMET 2MG/500MG 112 TABLETS **	
...	
A10BH Dipeptidyl peptidase 4 (DPP-4) inhibitors	
A10BH02 vildagliptin	1
*EUCREAS 50MG/850MG 60 TABLETS **	
...	
A10BX Other blood glucose lowering drugs, excl. insulins	

Continued on next page

Table 3.20 – continued from previous page

	min (mg**)
A10BX02 repaglinide	1.5
PRANDIN 2MG 90 TABLETS	
...	
C CARDIOVASCULAR SYSTEM	
C02 ANTIHYPERTENSIVES	
C02C ANTIADRENERGIC AGENTS, PERIPHERALLY ACTING	
C02CA Alpha-adrenoreceptor antagonists	
C02CA04 doxazosin	1
CARDURAN 8MG 28 TABLETS	
...	
C03 DIURETICS	
C03A LOW-CEILING DIURETICS, THIAZIDES	
C03AA Thiazides, plain	
C03AA03 hydrochlorothiazide	12.5
*COZAAR PLUS 50/12.5 28 COATED TABLETS **	
*CO-DIOVAN 160MG/12.5MG 28 FILM COATED TABLETS **	
*COAPROVEL 300/25MG 28 COATED TABLETS **	
*AMERIDE 5/50MG 60 TABLETS **	
*ENALAPRIL/HIDROCL BAYVIT 20/12.5MG 28 TABLETS **	
*ZESTORETIC 20/12.5MG 28 TABLETS **	
HIDROSALURETIL 50MG 20 TABLETS	
...	
C03B LOW-CEILING DIURETICS, EXCL. THIAZIDES	
C03BA Sulfonamides, plain	
C03BA04 chlortalidone	12.5
HIGROTONA 50MG 30 TABLETS	
...	
C03BA11 indapamide	1.25
TERTENSIF 1.5MG 30 COATED TABLETS	
...	
C03D POTASSIUM-SPARING AGENTS	
C03DB Other potassium-sparing agents	
C03DB01 amiloride	2.5
*AMERIDE 5/50MG 60 TABLETS **	
...	
C07 BETA BLOCKING AGENTS	
C07A BETA BLOCKING AGENTS	
C07AB Beta blocking agents, selective	
C07AB03 atenolol	25
ATENOLOL ALTER 50MG 60 TABLETS EFG	
...	
C07AB07 bisoprolol	2.5
EMCONCOR 5MG 30 COATED TABLETS	
...	
C07AB12 nebivolol	2.5
LOBIVON 5MG 28 TABLETS	
...	
C08 CALCIUM CHANNEL BLOCKERS	
C08C SELECTIVE CALCIUM CHANNEL BLOCKERS WITH MAINLY VASCULAR EFFECTS	

Continued on next page

Table 3.20 – continued from previous page

	min (mg**)
C08CA Dihydropyridine derivatives	
C08CA01 amlodipine	2.5
ASTUDAL 10MG 30 TABLETS	
*EXFORGE 10MG/160MG 28 FILM COATED TABLETS **	
...	
C08CA05 nifedipine	30
ADALAT OROS 30MG 28 TABLETS	
...	
C08CA08 nitrendipine	10
*ENEAS 10/20MG 30 TABLETS **	
...	
C08D SELECTIVE CALCIUM CHANNEL BLOCKERS WITH DIRECT CARDIAC EFFECTS	
C08DA Phenylalkylamine derivatives	
C08DA01 verapamil	120
*TARKA 180/2MG 28 TABLETS **	
...	
C09 AGENTS ACTING ON THE RENIN-ANGIOTENSIN SYSTEM	
C09A ACE INHIBITORS, PLAIN	
C09AA ACE inhibitors, plain	
C09AA01 captopril	25
CAPTOPRIL STADA 25MG 60 TABLETS EFG	
...	
C09AA02 enalapril	5
ENALAPRIL MERCK 20MG 28 TABLETS	
*ENALAPRIL/HIDROCL BAYVIT 20/12.5MG 28 TABLETS **	
*ENEAS 10/20MG 30 TABLETS **	
...	
C09AA03 lisinopril	5
LISINOPRIL MYLAN 20 MG 28 TABLETS	
*ZESTORETIC 20/12.5MG 28 TABLETS **	
...	
C09AA04 perindopril	2
COVERSYL 4MG 30 TABLETS	
...	
C09AA06 quinapril	5
ECTREN 20MG 28 COATED TABLETS	
C09AA10 trandolapril	0.5
*TARKA 180/2MG 28 CAPSULAS **	
...	
C09C ANGIOTENSIN II ANTAGONISTS, PLAIN	
C09CA Angiotensin II antagonists, plain	
C09CA01 losartan	25
COZAAR 100MG 28 COATED TABLETS	
*COZAAR PLUS 50/12.5 28 COATED TABLETS **	
...	
C09CA03 valsartan	80
DIOVAN 160MG 28 COATED TABLETS	
*CO-DIOVAN 160MG/12.5MG 28 FILM COATED TABLETS **	
*EXFORGE 10MG/160MG 28 FILM COATED TABLETS **	

Continued on next page

Table 3.20 – continued from previous page

	min (mg**)
...	
C09CA06 candesartan	8
PARAPRES 32MG 28 TABLETS	
...	
C09CA08 olmesartan medoxomil	10
IXIA 40MG 28 COATED TABLETS	
...	
Non-pharmacological	
Education	
Verification	
Check	
Follow-up	
Evaluation of risk factors	
Consultation	
Consultation endocrinology	
Complementary tests	
Laboratory	
ECG	
Radiology	
Other procedures	
Lower extremity oscillometry	
Tibial post. D(X)	

* Compound drugs

** UI/ml for A10A INSULINS AND ANALOGUES

4

Incremental generation of SDA diagrams with background knowledge

In this section we describe a procedure to generate SDA diagrams (see section 2.8) which considers all the relevant background knowledge introduced in chapter 3. Moreover, this procedure works incrementally and therefore, once a SDA diagram is generated for the first time, it can be updated as soon as new data arrives rather than being generated from scratch. The general scheme of the methodology presented is summarized in figure 4.1.

The data used to generate SDA diagrams is extracted from the *EOC database* which contains episodes of care, encounters, etc. of patients treated for a certain pathology, represented with the EOC data model (see section 2.7). This EOC database is created starting from a hospital database which uses a representation format as for example, HL7, ENV13606, OpenEHR, etc. A filtering and preprocessing procedure (which is not included in the scope of this thesis) is performed over this database in order to obtain an EOC database. Each time a new encounter is introduced in the hospital database, it is transformed to the EOC data model in order to generate a new SDA diagram.

The procedure needs some *background knowledge* in order to guarantee that the generated SDA diagram is in accordance with the medical knowledge which is not explicit in the EOC database. In this thesis, this background knowledge is obtained from the repository described in section 3.3. This repository can be filled with knowledge ex-

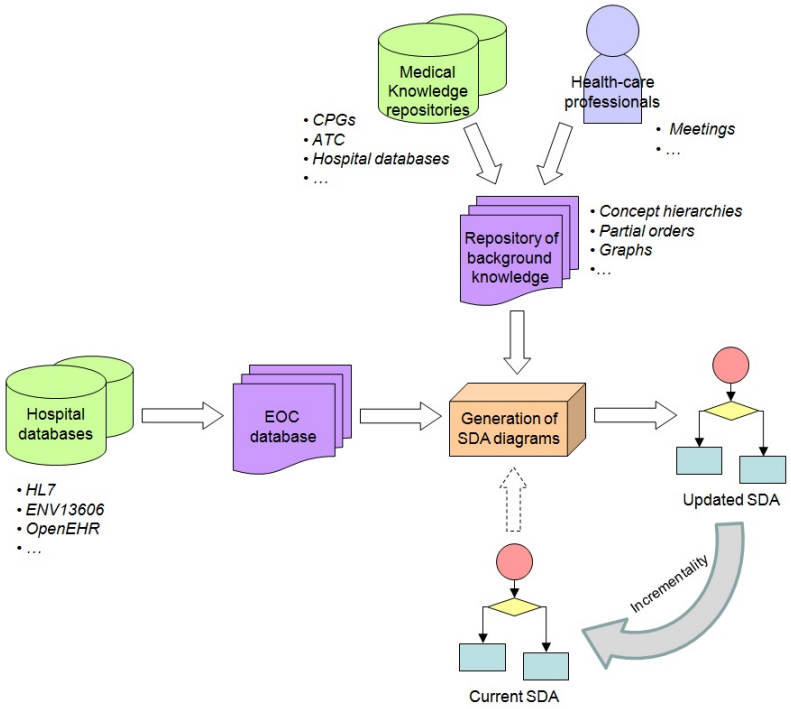


Figure 4.1: Scheme of the methodology to generate SDA diagrams

tracted from medical knowledge resources like CPGs (SAG02; SAG03), the Anatomical Therapeutic Chemical (ATC) Classification System (fDSM) or other hospital databases, or from health care professionals representing their preferences, experience, etc. Notice that each time that a SDA diagram is generated, the repository of background knowledge is consulted.

In addition to the patient data in the EOC database and the medical knowledge in the repository, the procedure may start from a previously generated SDA diagram. The first time that we generate a SDA diagram we do not have any previous one so it will be generated only using the EOC database and the background knowledge. The successive next times that we generate a new SDA diagram, the last SDA diagram obtained is used as an additional input of the SDA generation. This last SDA diagram is modified to incorporate the knowledge contained in the new encounters in the EOC database and the background knowledge, obtaining a new updated SDA diagram.

The procedure to generate SDA diagrams is divided into three steps each one solving a different problem (see figure 4.2). The first one solves the *identification of states*. The identification of states is the procedure used to identify the different health care states

(see definition 3.1.1) in which a certain patient may be located during the treatment of a disease. The set of states identified corresponds to the set of SDA states that the generated SDA diagram will have. In the successive incremental generations of the SDA diagram, the set of SDA states of the current SDA diagram is used as the starting point and it is updated according to the new patient data and medical knowledge.

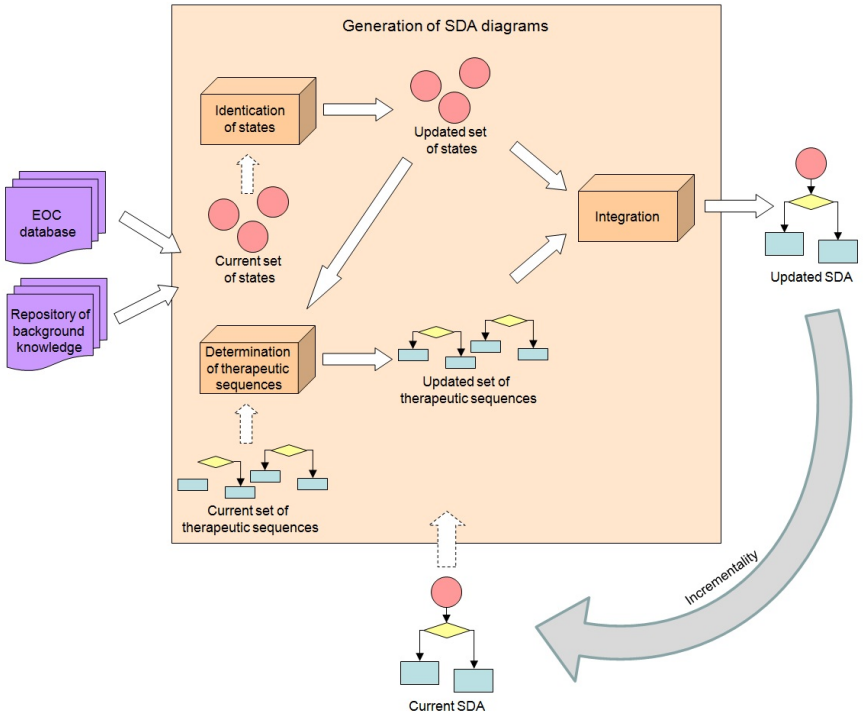


Figure 4.2: Scheme of the three steps to generate SDA diagrams

The second step is the *determination of therapeutic sequences* (see definition 3.1.5). The determination of therapeutic sequences is the procedure used to induce a sequence of concatenated questions to condition the sort of treatment to be followed for the patients evolving from a certain health care state to any other states. This corresponds to a sequence of connected SDA decisions that lead to different SDA actions, representing the treatment followed when evolving from one state to one or more next states. In the successive incremental generations of the SDA diagram, the therapeutic sequences in the current SDA diagram are used as baseline and they are updated considering the new patient data and medical knowledge.

Finally, the third step is the *integration*. In this step the resulting SDA diagram

is built by connecting the SDA states with the SDA decisions and the SDA actions obtained from the previous two steps.

4.1 Identification of states

The identification of states is the procedure used to identify the different health care states (see definition 3.1.1) in which a certain patient may be located during the treatment of a pathology. In the SDA model, health care states are represented as SDA states which are subsets of SDA state terms (see section 2.8).

We want to identify a set of states such that it has a maximum quality and medical sense, so we will start defining the concepts of quality of a state (section 4.1.1) and medical sense of a state (section 4.1.2). Then these two concepts will be used in an incremental algorithm to identify the set of states (section 4.1.3).

4.1.1 The quality of a state

The definition 3.1.1 of health care state expresses that a state must represent a *significant* group of patients that deserve a *particular course of action* and that this state must have some *interest* for the health care professional. Therefore, the quality of a SDA state involves three points of view which are classified as *epidemiological*, *therapeutic* and *preferential*. Each one of these views provides an answer to a different question:

- Epidemiological: Is this state representing a significant number of patients?
- Therapeutic: Are the treatments followed by the patients represented by the state homogeneous?
- Preferential: Does the health care expert consider this state relevant?

The epidemiological view is related to the significance of the set of encounters represented by this state. The epidemiological quality of a state can be calculated then as the number of encounters represented by this state divided by the total number of episodes. Given a state S_i we denote $\epsilon(S_i)$ the proportion of encounters that S_i represents.

The therapeutic view is related to the homogeneity of the treatments proposed for the patients in this state. Given a state S_i, let $\epsilon(S_i)$ be the set of encounters of patients being in state S_i and $A(\epsilon(S_i))$ the multiset containing the clinical action performed in each $\epsilon(S_i)$, we calculate the treatment homogeneity $h(A(\epsilon(S_i)))$ of the state S_i with the procedure described in section 3.1.5.

The preferential view is related to the relevance of the state according to the health care expert preferences. To represent these preferences we use a partial order $\leq_S$ on the state terms (see section 3.1.2). This LPO can be then transformed into a preference function $p : S \rightarrow [0, 1]$ preserving the information provided by the LPO (LVR11). This is formalized by the properties of *normality* (i.e., $\forall s \in S, \{0 \leq p(s) \leq 1\}$) and *monotonicity* (i.e., $\forall s_1, s_2 \in S$, $\{s_1 \leq_S s_2 \Rightarrow p(s_1) \leq p(s_2)\}$). As $\leq_S$ does not provide information about the distance between elements, we assume that the distance between consecutive layers of the LPO is constant, we call this the *equidistance* property. So, if $\leq_S$ has n layers and $\ell(s)$ is the layer where the element s is, equation 4.1 defines the only preference function p which preserves the partial order of $\leq_S$ for all the values $s \in S$ and which satisfies normality, monotonicity and equidistance properties.

$$p(s) = \begin{cases} \dfrac{\ell(s) - 1}{n - 1} & \text{if } n > 1 \\ 1 & \text{otherwise} \end{cases} \tag{4.1}$$

Finally, the preferential quality $p(S_i)$ of a state S_i is computed as the average of the preference values of the state terms in S_i (i.e., $p(S_i) = \sum_{s \in S_i} p(s)/|S_i|$).

Given a state S_i, we calculate the quality of this state with the function $quality(S_i)$ which can be equal to $\epsilon(S_i)$, $h(A(\epsilon(S_i)))$ or $p(S_i)$ depending on whether we want an epidemiological, therapeutic or preferential view. Notice that the epidemiological view has a bias towards the states with less state terms while the therapeutic view has a bias towards the states with more state terms. Another approach is to combine the three points of view giving a certain weight α to each of them (i.e., $quality(S_i) = \alpha_e \epsilon(S_i) + \alpha_t h(A(\epsilon(S_i))) + \alpha_p p(S_i)$).

4.1.2 The medical sense of a state

In spite of the epidemiological, therapeutic or preferential quality of health care states, they must have a *medical sense*. The medical sense depends on both the coherence of the description of the state and the coherence of the state itself with respect to

the rest of states as it is explained in section 3.1.1. The knowledge used to guarantee medical sense is represented with a state constraints graph G_S (see definition 3.1.3). This background knowledge is necessary to assure that, on the one hand, given a SDA state its terms are not redundant or medically incorrect and, on the other hand, all the SDA states in a SDA diagram are defined at the same level of abstraction of the medical terminology. Therefore, although the state constraints graph contains one type of constraint, these constraints are used to guarantee two aspects: the medical sense within each state and the medical sense of the whole set of states.

In order to incorporate the constraints in G_S to the identification of states, we distinguish between: *intra-state constraints* and *inter-state constraints*.

Definition 4.1.1 (Intra-state constraint) *Each constraint $c = \{s_i, s_j\}$ in the state constraints graph G_S represents an intra-state constraint, meaning that the state terms s_i and s_j cannot be in the same state (i.e., we do not allow a state S_i such that $s_i, s_j \in S_i$ to be in the SDA diagram).*

Definition 4.1.2 (Inter-state constraint) *Each constraint $c = \{s_i, s_j\}$ in the state constraints graph G_S represents an inter-state constraint, meaning that the states S_i and S_j cannot be in the same SDA diagram if $s_i \in S_i$ and $s_j \in S_j$ (i.e., if a state S_i that contains s_i is included in a SDA diagram, we cannot include a state S_j that contains s_j in the same SDA diagram, and vice versa).*

For example, considering the state constraints graph for hypertension in table 3.7, there is a constraint between the terms HEART RISK and LVH. Therefore, this constraint will represent an intra-state constraint that will not allow these two terms to be in a same state; and a inter-state constraint that will not allow two states to be in the same diagram if they contain the terms HEART RISK and LVH respectively.

4.1.3 State identification algorithm

The identification of states is solved as a clustering problem (see section 2.1). The encounters of the EOC database have to be assigned a cluster (i.e., a health care state). According to the type-0 non-determinism of the SDA model (see section 2.8), the two sets of patients that satisfy two health care states may intersect, thus we are dealing with an overlapping clustering problem (see section 2.3). Moreover, we must assure the medical sense of the health care states by means of constrains so it is also a constrained

clustering problem (see section 2.5.2), and the constant arrival of new data in the EOC database suggests that the clustering must be done incrementally (see section 2.6.1). Nevertheless, the existing clustering methods are not useful to identify health care states because the quality of a set of health care states depends on criteria which cannot be derived from the state terms.

The problem of state identification is formally defined as the following. Given a set of encounters of the EOC database $E = \{enc_1, ..., enc_n\}$, each encounter is a meeting patient-physician where the physician observes a set of characteristics $S(enc_i)$, represented as state terms, that describe the state of the patient at this moment; and proposes some health care measures. We generate the best set of clusters $\mathcal{S} = \{S_1, ..., S_k\}$, where k is not given a priori and each cluster $S_i \in \mathcal{S}$ represents a SDA state $S_i = \{s_1, ..., s_j\}, s_i \in S$. If $S(enc_i) \subseteq S_i, S_i \in \mathcal{S}$ then this encounter is represented by this SDA state. Notice that one encounter can be represented by more than one SDA state. The quality of a set of states is calculated as the average of the qualities $quality(S_i)$ of each state S_i, where *quality* is implemented as it is explained in section 4.1.1. There are five constraints that we want the set of states generated to fulfill:

1. The number of terms in a state must be greater or equal than a constant $minS$ ($\forall S_i \in \mathcal{S}, |S_i| \geq minS$) in order to avoid diagrams with too less states and which are too general.

2. The number of terms in a state must be lower or equal than a constant $maxS$ ($\forall S_i \in \mathcal{S}, |S_i| \leq maxS$) in order to avoid diagrams with too much states and which are too specific.

3. A health care state must represent one or more encounters ($\forall S_i \in \mathcal{S}, |\epsilon(S_i)| > 0$) in order to be representative enough to be considered for the diagram.

4. States having intra-state constraints between some of their terms are not allowed (given a state constraints graph $G_S = (S, C)$ and $(s_i, s_j) \in C$, then $s_i, s_j \in S_i \Rightarrow S_i \notin \mathcal{S}$)

5. States having inter-state constraints between them are not allowed (given a state constraints graph $G_S = (S, C)$, $(s_i, s_j) \in C$, $s_i \in S_i$ and $s_j \in S_j$, then $S_i \in \mathcal{S} \Rightarrow S_j \notin \mathcal{S}$ and $S_j \in \mathcal{S} \Rightarrow S_i \notin \mathcal{S}$)

4. INCREMENTAL GENERATION OF SDA DIAGRAMS WITH BACKGROUND KNOWLEDGE

Starting from the EOC database we propose two algorithms to solve the identification of the states: the *non-incremental identification* algorithm and the *incremental identification* algorithm. If we do not have previously identified states, we apply the non-incremental identification algorithm. This happens the first time that we identify the states of a SDA. Once a set of states exists, we apply the incremental algorithm to incorporate the information contained in new encounters to the previous SDA.

Both algorithms are explained in the next paragraphs.

Non-incremental identification algorithm: Algorithm 1 shows the procedure followed in the non-incremental identification of states.

This algorithm receives as input a set of encounters E_{PC}. The encounters of this set only contain the patient condition, that is to say the set of state terms (see section 2.7) because decision and action terms are not needed. We omit showing the different background knowledge structures and functions used in the algorithm. It starts by generating the space of possible states $\mathcal{S}'$ making all the combinations of state terms $s \in S$. $\mathcal{S}'$ will contain a state for each one of the subsets of S except the empty set (i.e., $\mathcal{P}(S) - \{\emptyset\}$). We can restrict the minimal and maximal number of terms in the states of $\mathcal{S}'$ with the parameters $minS$ and $maxS$, respectively (i.e., $\mathcal{S}' = s \in P(S) : minS \leq |s| \leq maxS - \emptyset$). This process uses the intra-state constraints of the state constraints graph $G_S = (S, C)$ to avoid states having incompatible terms $s_i, s_j, ...$ such that $(s_i, s_j) \in C$. Each encounter in E_{PC} describes a patient that is in each one of the states in $\mathcal{S}'$ whose terms are all observed for that patient in that encounter. If $\epsilon(S_i)$ is the set of encounters represented by a certain state S_i (i.e., $\epsilon(S_i) = \{enc_j \in E_{PC}, S(enc_j) \subseteq S_i\}$), then $\mathcal{S}'$ does not contain states s such that $epsilon(s) = \emptyset$ (i.e., $\mathcal{S}'s \in P(S) : \epsilon(s) = \emptyset = \emptyset$). A state S_i which does not represent any encounter in E_{PC} ($\epsilon(S_i) = \emptyset$) is not included in $\mathcal{S}'$. So, the final number of states in $\mathcal{S}'$ is $\sum_{i=minS}^{maxS} \binom{|S|}{i}$ less the number of states having an intra-state constraint $|\{S_i \subseteq \mathcal{P}(S) : \exists(s_i, s_j) \in C, s_i, s_j \in S_i\}|$ and less the number of states that do not represent any encounter $|\{S_i \subseteq \mathcal{P}(S) : \epsilon(S_i) = \emptyset\}|$.

In order to show an example of the construction of a space of states, consider a set of encounters E_{PC} that contains these following combinations of state terms about one or more patients in states e_1, e_2, e_3, or e_4, with:

- e_1: *SMOKES, HEART RISK, S1 HYPERTENSIVE SBP, HIGH CREATININE*

Algorithm 1: non-incremental_identification_of_states

Input: E_{PC}: set of encounters (only their patient condition)

Output: $\mathcal{S}$: set of identified states, $\mathcal{S}''$: initial space of states, $qualities : double[]$

```
1  S' ← generate_space_of_states(E_PC, minS, maxS, C);
2  qualities : double[|S'|];
3  foreach S_i ∈ S' do
4      qualities[S_i] = calculate_quality(S_i);
5  S ← ∅;
6  represented ← ∅;
7  S'' ← S';
8  while S' ≠ ∅ do
9      best = arg max_{S_i ∈ S'} qualities[S_i];
10     S ← S ∪ best;
11     represented ← represented ∪ ε(best);
12     foreach S_i ∈ S' do
13         if (ε(S_i) ⊆ represented) OR
           (∃{s_i, s_j} ∈ C : s_i ∈ S_i ∧ s_j ∈ S_j ∧ S_j ∈ represented) then
14             S' ← S' − S_i;
15 return S, S'', qualities;
```

- e_2: *DOES NOT SMOKE, NO HEART RISK, NORMAL SBP*
- e_3: *DOES NOT SMOKE, NO HEART RISK, PREHYPERTENSIVE SBP*
- e_4: *SMOKES, HEART RISK, PREHYPERTENSIVE SBP*

Then, figure 4.3 depicts the space of states $\mathcal{S}'$ for $minS = 1$ and $maxS = 3$ and considering the set of intra-state constraints represented in the state constraint graph of the case is $C = \{$*(SMOKES, NO HEART RISK), (DOES NOT SMOKE, HEART RISK), (DOES NOT SMOKE, NOT HEART RISK)*$\}$.

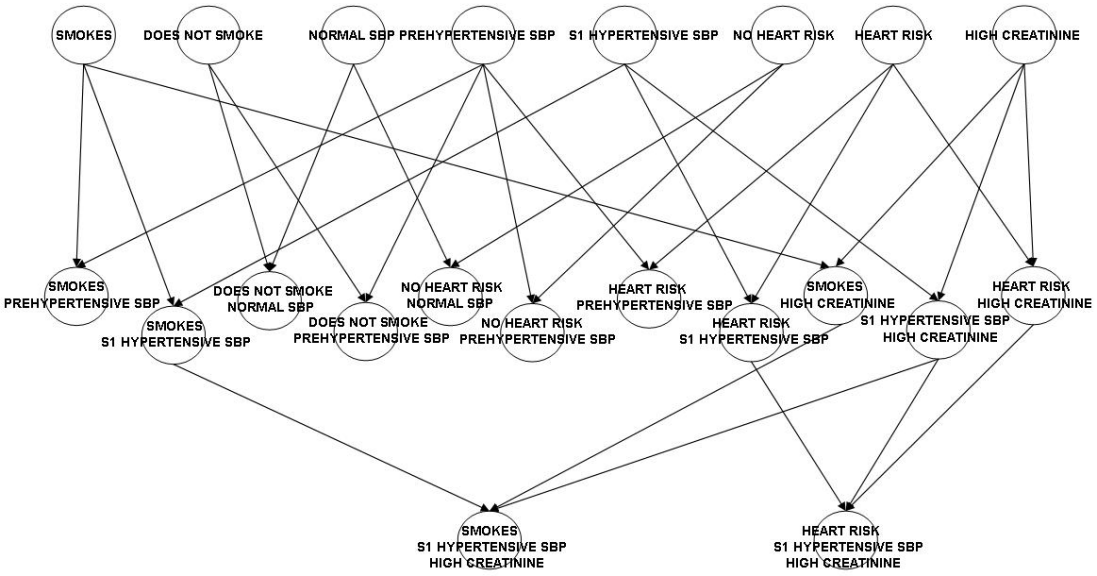


Figure 4.3: Example of space of states

For each state $s \in \mathcal{S}'$ the algorithm calculates the quality of this state $quality(s)$ (lines 2-4). The quality of a state can be calculated with epidemiological, therapeutic and preferential approaches as it is proposed in section 4.1.1. Finally, we repeat the removal of states in $\mathcal{S}'$ until $\mathcal{S}'$ is empty. The removal criteria are:

1. Remove from $\mathcal{S}'$ the state with the greatest quality, at each loop.
2. Remove from $\mathcal{S}'$ all the states representing encounters which are represented by one of the identified states.
3. Remove from $\mathcal{S}'$ all the states having terms with an inter-state constraint with any term in one of the identified states.

Algorithm 1 returns $\mathcal{S}$ a set of states which is representative of the patient states in the encounters E_{PC}, a copy of the initial space of states $\mathcal{S}''$ and the qualities of each state in $\mathcal{S}''$.

Following the previous example, in table 4.1 we specify for each state in the initial space of states $\mathcal{S}'$ (column 2), the kinds of encounters that it represents (column 3) and a possible quality value (column 4). The states are given an id number (column 1) and they are ordered according to their quality. The identified states are in bold and, for each of them, we specify the id number of the states removed. The first identified state (the one with the highest quality) is *HEART RISK,S1 HYPERTENSIVE SBP*. It removes all the states representing only e_1 (see last column). We suppose that there is a inter-state constraint between *HEART RISK* and *SMOKES*, and between *HEART RISK* and *DOES NOT SMOKE* so other states like *SMOKES,PREHYPERTENSIVE SBP* are also removed. The other identified states are *NORMAL SBP* and *PREHYPERTENSIVE SBP* which also remove some states.

Incremental identification algorithm: Algorithm 2 shows the procedure followed in the incremental identification of states.

The algorithm receives as input a set of new encounters E_{PC}, as well as the three output parameters of the previous identification procedure ($prev\_\mathcal{S}$, $prev\_\mathcal{S}'$ and $prev\_qualities$). We omit showing the different background knowledge structures and functions used in the algorithm. First of all, it extends the space of states with the new states obtained from the new encounters, if there are any (line 1).

Consider the previous example and suppose that there is a new state:

- e_5: *DOES NOT SMOKE, NO HEART RISK, HYPOTENSIVE SBP*

The previous space of states $\mathcal{S}'$ is extended with states: *HYPOTENSIVE SBP*, *DOES NOT SMOKE,HYPOTENSIVE SBP* and *NO HEART RISK, HYPOTENSIVE SBP*.

Then, the quality values of the states representing any new encounter are updated (lines 3-5). Then it keeps two sets that contain the only states that must be revised (at the moment). The first set is called *revise* and it initially contains the states that have advanced other states in terms of quality with respect to the last identification. $(S_i \in revise \Leftrightarrow \exists S_j, (prev\_qualities[S_j] > prev\_qualities[S_i]) \wedge (qualities[S_j] < qualities[S_i]))$ (line 6). The other set is called *revise_encounters* and it contains the

Id	State	Represents	Quality	Removes
0	***HEART RISK,S1 HYPERTENSIVE SBP***	e_1	0.90	1,3,4,5,8,11,13,14,15,17,18,19,20
1	*S1 HYPERTENSIVE SBP*	e_1	0.87	
2	***NORMAL SBP***	e_2	0.85	10
3	*HIGH CREATININE*	e_1	0.81	
4	*S1 HYPERTENSIVE SBP,HIGH CREATININE*	e_1	0.80	
5	*HEART RISK,HIGH CREATININE*	e_1	0.78	
6	***PREHYPERTENSIVE SBP***	e_3,e_4	0.77	7,9,12,16
7	*HEART RISK,PREHYPERTENSIVE SBP*	e_4	0.75	
8	*HEART RISK,S1 HYPERTENSIVE SBP,HIGH CREATININE*	e_1	0.72	
9	*HEART RISK*	e_1,e_4	0.70	
10	*NO HEART RISK,NORMAL SBP*	e_2	0.68	
11	*SMOKES,S1 HYPERTENSIVE SBP*	e_1	0.66	
12	*NO HEART RISK,PREHYPERTENSIVE SBP*	e_3	0.65	
13	*SMOKES,HIGH CREATININE*	e_1	0.62	
14	*SMOKES,PREHYPERTENSIVE SBP*	e_4	0.61	
15	*SMOKES,S1 HYPERTENSIVE SBP,HIGH CREATININE*	e_1	0.60	
16	*NO HEART RISK*	e_2, e_3	0.51	
17	*DOES NOT SMOKE,PREHYPERTENSIVE SBP*	e_3	0.44	
18	*SMOKES*	e_1, e_4	0.43	
19	*DOES NOT SMOKE,NORMAL SBP*	e_2	0.41	
20	*DOES NOT SMOKE*	e_2, e_3	0.32	

Table 4.1: Example of ranking of states according to their quality during the identification process

Algorithm 2: incremental_identification_of_states

Input: E_{PC}: set of encounters (only their patient condition), $prev\_\mathcal{S}$: set of previously identified states, $prev\_\mathcal{S}'$: previous space of states, $prev\_qualities$: array of previous qualities

Output: $\mathcal{S}$: set of identified states, $\mathcal{S}''$: initial space of states, $qualities$: $double[]$

```
1  S' ← prev_S' + determine_new_states(E_PC, minS, maxS, C);
2  qualities : double[|S'|];
3  foreach S_i ∈ S' do
4      if represents_new_encounters(S_i) then qualities[S_i] = calculate_quality(S_i);
5      else qualities[S_i] = prev_qualities[S_i];
6  revise ← advancing_states(S', prev_qualities, qualities);
7  revise_encounters ← prev_S' − revise;
8  S ← ∅;
9  represented ← ∅;
10 foreach S_i ∈ revise ∪ revise_encounters in descendant order of quality(S_i) do
11     if S_i ∈ revise_encounters then
12         S ← S ∪ S_i;
13         represented ← represented ∪ ε(S_i);
14     else
15         if ε(S_i) ⊈ represented then
16             S ← S ∪ S_i;
17             represented ← represented ∪ ε(S_i);
18             revise ← revise ∪ identified_advanced(S_i);
19             if S_i ∉ prev_S then revise ← revise ∪ identified_behind(S_i);
20         else
21             if S_i ∈ prev_S then revise ← revise ∪ not_identified_behind(S_i);
22         revise_encounters ← revise_encounters − revise;
23 return S, S', qualities;
```

previously identified states not included in *revise* (i.e., $revise\_encounters = prev\_\mathcal{S} - revise$) (line 7).

Each one of the states S_i in these sets is revised with the following procedure starting from the one with a highest quality. If $S_i \in revise\_encounters$, we identify this state (lines 12-13). Otherwise, the state is identified if it represents patient states in encounters which have not been represented by any encounters of any identified state ($\epsilon(S_i) \not\subseteq represented$) (lines 16-17). If S_i is identified, there is an additional set of states that must be revised. These states were identified in the previous identification and now they may be not identified because of S_i. This set of states is called $identified\_advanced(S_i)$ (line 18) and contains all the states S_j that fulfill:

- They were identified in the previous identification ($S_j \in prev\_\mathcal{S}$).
- They had a quality greater than S_i in the previous identification ($prev\_qualities[S_j] > prev\_qualities[S_i]$).
- They have a quality lower than S_i in the current identification ($qualities[S_j] < qualities[S_i]$).

Moreover, if S_i was not identified in the previous identification ($S_i \notin prev\_\mathcal{S}$), we must also revise the states in $identified\_behind(S_i)$ (line 19) which are the states S_j that fulfill:

- They were identified in the previous identification ($S_j \in prev\_\mathcal{S}$).
- They have a quality lower than S_i in the current identification ($qualities[S_j] < qualities[S_i]$).

In case that S_i is not identified in the current procedure, if S_i was identified in the previous identification ($S_i \in prev\_\mathcal{S}$) there is another set of states that must be revised. These states were not identified in the previous identification maybe because of S_i. Now that S_i is not identified, their situation may change. This set of states is called $not\_identified\_behind(S_i)$ (line 21) and contains all the states S_j that fulfill:

- They were not identified in the previous identification ($S_j \notin prev\_\mathcal{S}$).
- They have a quality lower than S_i in the current identification ($qualities[S_j] < qualities[S_i]$).

Finally, the new states in *revise* are removed from *revise_encounters* (line 22). The algorithm returns the set of identified states $\mathcal{S}$, the initial space of states $\mathcal{S}'$ and the qualities of each state in $\mathcal{S}'$.

Following the same example than before, table 4.2 contains the updated ranking of states. The column +/- is used to mark the states that have advanced other states ('+') and those that have been advanced ('-'). Finally, a '*' mark in column *Revised* means that this state will be revised during the identification process.

Initially, the set *revise* contains the states *HEART RISK*, *HYPOTENSIVE SBP*, *NO HEART RISK, HYPOTENSIVE SBP*, and *DOES NOT SMOKE,HYPOTENSIVE SBP*. The set *revise_encounters* contains *HEART RISK,S1 HYPERTENSIVE SBP*, *NORMAL SBP* and *PREHYPERTENSIVE SBP*.

We start revising the state *HEART RISK,S1 HYPERTENSIVE SBP* which is in *revise_encounters* so it is automatically identified. The same happens with the next state *NORMAL SBP*. The third state to be revised is *HEART RISK* which is not in *revise_encounters*. In this case we identify it because it represents the encounters of the kind e_4, which are not represented by the previous identified states. Then we must also revise $identified\_advanced(HEART\ RISK)$ which contains the state *PRE-HYPERTENSIVE SBP*, and also $identified\_behind(HEART\ RISK)$ which in this case is empty. Finally we remove *PREHYPERTENSIVE SBP* from *revise_encounters*. The next state to revise is *PREHYPERTENSIVE SBP* which is identified again. The sets $identified\_advanced(PREHYPERTENSIVE\ SBP)$ and *identified_behind (PREHY-PERTENSIVE SBP)* are empty. The last states to be revised are the three new states. The first one is *HYPOTENSIVE SBP* which is identified. The set *identified_advanced (HYPOTENSIVE SBP)* is empty but the set $identified\_behind(HYPOTENSIVE\ SBP)$ contains all the states that have been advanced by this new state, and have to be revised. All the remaining states are not identified. So with this new incremental identification of states, two additional states have been identified. Notice that only half of the states have been revised.

The incremental identification of states allows us to deal more efficiently with the constant arrival of data of new encounters because for each new identification of states we do not have to create the whole space of possible states from scratch but to add the new ones. Likewise, only the quality of states involving new encounters has to be recalculated. Another advantage of this incremental solution is that the proposed algorithm

Id	State	Represents	Quality	+/-	Revised	Removes
0	***HEART RISK,S1 HYPERTENSIVE SBP***	e_1	0.90		*	1,3,4,5,9,11,13,14,15,18,19,21,22,23
1	*S1 HYPERTENSIVE SBP*	e_1	0.87			
2	***NORMAL SBP***	e_2	0.85		*	10
3	*HIGH CREATININE*	e_1	0.81			
4	*S1 HYPERTENSIVE SBP,HIGH CREATININE*	e_1	0.80			
5	*HEART RISK,HIGH CREATININE*	e_1	0.78			
6	***HEART RISK***	e_1,e_4,	0.76	+	*	8
7	***PREHYPERTENSIVE SBP***	e_3,e_4	0.74	-	*	12
8	*HEART RISK,PREHYPERTENSIVE SBP*	e_4	0.70	-		
9	*HEART RISK,S1 HYPERTENSIVE SBP,HIGH CREATININE*	e_1	0.69	-		
10	*NO HEART RISK,NORMAL SBP*	e_2	0.68			
11	*SMOKES,S1 HYPERTENSIVE SBP*	e_1	0.66			
12	*NO HEART RISK,PREHYPERTENSIVE SBP*	e_3	0.65			
13	*SMOKES,HIGH CREATININE*	e_1	0.62			
14	*SMOKES,PREHYPERTENSIVE SBP*	e_4	0.61			
15	*SMOKES,S1 HYPERTENSIVE SBP,HIGH CREATININE*	e_1	0.60			
16	***HYPOTENSIVE SBP***	e_5	0.57	+	*	17,20
17	*NO HEART RISK*	e_2, e_3, e_5	0.51	-	*	
18	*DOES NOT SMOKE,PREHYPERTENSIVE SBP*	e_3	0.44	-	*	
19	*SMOKES*	e_1, e_4	0.43	-	*	
20	*NO HEART RISK, HYPOTENSIVE SBP*	e_5	0.42	+	*	
21	*DOES NOT SMOKE,NORMAL SBP*	e_2	0.41	-	*	
22	*DOES NOT SMOKE,HYPOTENSIVE SBP*	e_5	0.38	+	*	
23	*DOES NOT SMOKE*	e_2, e_3, e_5	0.32	-	*	

Table 4.2: Example of ranking of states according to their quality during the incremental identification process

just revises the minimum subset of states that is necessary to guarantee a correct identification. Moreover, the set of identified states only depends on the encounters used, without regard to the sequence in which those encounters were presented. As far as memory requirements is concerned, the proposed solution, instead of keeping the whole set of encounters for each incremental identification, it only stores one encounter for each combination of state terms together with an identifier. Each state has a list with the identifiers of the encounters that it represents and the amount of them, avoiding the storage of redundant data.

4.2 Determination of therapeutic sequences

The determination of therapeutic sequences (see definition 3.1.5) is the procedure used to induce a sequence of questions to determine the sort of treatment to be followed for the patients with a certain health care state.

In the SDA model, questions are represented as SDA decisions that allow the integration of all the variability that a treatment may have by means of conditions on several SDA decision terms representing available information about the patient and the patient's health care condition. The variability of a treatment is represented with SDA actions, which are subsets of SDA action terms, constituting the proper health care activities involved in the health care procedure represented (see section 2.8).

We want to induce a therapeutic sequence such that it is medically comprehensible and correct (LVRB12). We will start defining the concepts of comprehensibility (section 4.2.1) and correctness (section 4.2.2) of a therapeutic sequence. In section 4.2.3 these two concepts will be used in an incremental algorithm to induce comprehensible and correct therapeutic sequences.

4.2.1 Comprehensibility of a therapeutic sequence

Comprehensibility (LVRB12) is a measure of the adherence to the order followed by a physician when gathering evidences (represented as SDA decisions) in a therapeutic sequence. In the induction of therapeutic sequences, comprehensibility is guaranteed by means of a partial order $\leq_D$ (see section 3.1.4) on the decision terms D. This partial order represents the background knowledge available about what are the questions that should be asked before which other questions in a therapeutic sequence. This knowledge

should be provided by a health care expert considering several health care criteria such as the order suggested in CPGs, care preferences or past experiences (LVR12). This background knowledge should be considered during the induction of therapeutic sequences in order to foster that decisions with a higher priority appear before in therapeutic sequences.

4.2.2 Correctness of a therapeutic sequence

Correctness (LVRB12) is a measure of how relevant are the treatment errors made when a SDA action is decided at the end of a therapeutic sequence. In the induction of therapeutic sequences, correctness is guaranteed by means of a similarity function s (see section 3.1.5). This similarity function represents the background knowledge that is used to calculate the medical homogeneity h of a multiset of actions. During the induction of therapeutic sequences, we calculate the homogeneity between the treatments that received a certain set of patients. If it is homogeneous enough, there is no need for more questions (i.e., SDA decisions) and a single global treatment (i.e., SDA action) can be given.

4.2.3 Therapeutic sequences induction algorithm

The induction of a therapeutic sequence can be seen as a decision tree induction problem (see section 2.4). Considering the set of encounters in the EOC database of patients in a certain health care state, a Decision Tree (DT) can be induced such that the questions are decision terms and the final decisions are sets of action terms. This sort of DT uses decision terms to discriminate between the sorts of treatments that have been prescribed in these encounters. This DT can be transformed into a therapeutic sequence represented with the SDA model replacing the internal nodes with SDA decisions and the leaves with SDA actions. We can guarantee medical comprehensibility and correctness of the therapeutic sequence with cost-sensitive learning (see section 2.5.3). The constant arrival of new data in the EOC database suggests that the induction process must be incremental (see section 2.6.2).

In order to accomplish all the above mentioned requirements, we propose an incremental DT induction algorithm which is based on the ITI family of algorithms (Utg88; Utg89; UBC97) with an alternative measure to choose questions and decisions which takes into account the background knowledge of the domain.

The problem of induction of a therapeutic sequence is formally defined as it follows. Given a SDA state S_1, we consider $\epsilon(S_1)$ the set of encounters in the EOC database of patients in state S_1. Starting from $\epsilon(S_1)$, we induce the most medically comprehensible and correct DT that classifies each encounter $enc_i \in \epsilon(S_1)$ according to their action terms $A(enc_i)$, making partitions using sets of decision terms $D(enc_i)$.

In order to generate these DTs, we use sets of encounters E_{HC} where each encounter only contains health care measures; that is to say, decision and action terms. State terms are no needed. The incremental approach presented in this section uses the storage of counters instead of sets of encounters E_{HC} within the internal nodes of the decision trees. The counters are used to avoid the redundant storage of encounters in the DT. When generating DTs, instead of maintaining the corresponding sets of encounters in each node (which is the usual approach), we maintain counters. In figure 4.4(a) we can see an example of a set of encounters used in the induction of therapeutic sequences for the domain of Hypertension (HT). Each row represents a different encounter and contains its decision terms (column 1) and its actions (column 2), which can be {*ALCOHOL, NOT ALCOHOL, SMOKES, DOES NOT SMOKE*} and {*EDUCATION, ENALAPRIL MERCK 20MG 28 TABLETS*}, respectively. As this way of representation maintains all the different combinations of decision and action terms it can imply a huge amount of memory for large sets of encounters. Figure 4.4(b) depicts the same example represented by means of two counters (one for each semantic decision which are called *Alcohol* and *Smoking*). Each counter has one column for each action, and one row for each decision term in the semantic decision plus the *otherwise* alternative if none of the decision terms appear in the encounter. The counters contain the number of encounters having each action while having each decision term (or not). With this representation the amount of memory used by the decision trees does not grow with the number of encounters and we still keep all the information needed to induce the therapeutic sequences. The sets of encounters E_{HC} are only kept in the leaves of the DT, where they are needed in order to expand the leaves if it is necessary.

Most of the algorithms to induce DTs have three key points in which a decision has to be made according to one or more criteria (LVRB12). Here we call them *choice points* and these are the following:

- Condition for placing a decision node (or not): If this condition is fulfilled the

ALCOHOL, SMOKES	**EDUCATION**
ALCOHOL, NOT FOLLOWING DIET	**ENALAPRIL MERCK...**
SMOKES	**EDUCATION**
ALCOHOL, SMOKES, NOT FOLLOWING DIET	**EDUCATION**
NOT FOLLOWING DIET	**ENALAPRIL MERCK...**
SMOKES	**ENALAPRIL MERCK...**
SMOKES, NOT FOLLOWING DIET	**EDUCATION**
ALCOHOL, NOT FOLLOWING DIET	**ENALAPRIL MERCK...**
ALCOHOL, SMOKES	**EDUCATION**
SMOKES	**ENALAPRIL MERCK...**
ALCOHOL, NOT FOLLOWING DIET	**EDUCATION**
ALCOHOL, SMOKES	**EDUCATION**

(a) An example of set of encounters

	Alcohol	
	EDUCATION	ENALAPRIL MERCK ...
ALCOHOL	4	1
NOT ALCOHOL	1	3
otherwise	2	1

	Smoking	
	EDUCATION	ENALAPRIL MERCK ...
SMOKES	5	1
DOES NOT SMOKE	1	3
otherwise	1	1

(b) An example of counters

Figure 4.4: Sets of encounters *vs* counters

algorithm places a decision node and otherwise it keeps placing question nodes to partition the set of encounters.

- Selection of the best decision: A certain measure is used to decide which one of the possible decisions is the best for the current set of encounters.
- Selection of the best question: A certain measure is used to decide which one of the remaining questions is the best to partition the current set of encounters.

Considering these three choice points, we define three functions that will be used in the algorithm to induce a therapeutic sequence. These are the functions similar_action, best_action and best_decision. They are described in detail in the next paragraphs. Notice that these functions work both with counters or sets of encounters, however we always refer to sets of encounters E_{HC} to clarify the explanation:

***similar_action*(E_{HC})**

During the induction of a DT, when each one of the encounters in a set has a similar action, a final decision can be made proposing a common action for all these encounters. The boolean function $similar\_action(E_{HC})$ is true when all the encounters in E_{HC} have a similar action. In this case, the condition for making a decision on the medical action is fulfilled. When a final decision is made over a set of encounters, this decision may not be completely correct from a medical point of view for each one of the encounters. In order to guarantee the medical correctness of the therapeutic sequence induced,

the medical errors committed must not be relevant (see section 4.2.2). Therefore, the function $similar\_action(E_{HC})$ uses the background knowledge of the similarity function s to determine how similar are the different actions in the encounters of E_{HC}. Concretely, it calculates the homogeneity $h(A(E_{HC}))$ of the treatments in the set of encounters E_{HC} (see section 3.1.5). If and only if the homogeneity is greater than a predefined similarity threshold δ, then $similar\_action(E_{HC})$ is true.

best_action(E_{HC})

Given a set of encounters E_{HC}, the function $best\_action(E_{HC})$ provides the best action to make (i.e., the most correct action from a medical point of view). In order to guarantee medical correctness (see section 4.2.2), the function chooses the action with a higher average similarity with all the actions in the encounters of E_{HC}. For each action A_i contained in $A(E_{HC})$ (i.e., each action in some of the encounters in E_{HC}), we calculate $\frac{1}{\#E_{HC}} \sum_{enc_i \in E_{HC}} s(A_i, A(enc_i))$. The action obtaining a higher results is considered the best one.

best_decision(E_{HC},SD)

Given a set of encounters E_{HC} and a set of possible semantic decisions SD, the function $best\_decision(E_{HC}, SD)$ provides the best semantic decision in terms of medical comprehensibility, and which is also a useful decision to decide the final action (i.e., a decision that when partitions E_{HC} leads to a better situation to decide a final action for the therapeutic sequence). In order to select a decision that fulfills both points, the $best\_decision(E_{HC}, SD)$ function follows the next procedure. First of all, it chooses the most comprehensible semantic decisions (see section 4.2.1) using the decisions partial order $\leq_D$. This is done by selecting those semantic decisions which have a higher priority according to $\leq_D$ (i.e., we select the semantic decisions in $SD_1 = \{sd_i \in SD : \nexists sd_j \in SD, sd_i \neq sd_j | sd_j \leq_D sd_i\}$.

Then, for each one of the selected semantic decisions in SD_1, the function calculates the *expected homogeneity* (*eh*) (LVRB12). Given a semantic decision sd, $eh(sd, E_{HC})$ represents the average homogeneity of a pairwise comparison of the homogeneities of the treatments in the subsets of encounters obtained after partitioning E_{HC} with sd. This $eh(sd, E_{HC})$ value is useful to determine whether the use of sd at this point of the

therapeutic sequence leads to a better situation to decide a final action, or not. eh is calculated with equation 4.2, where $E_d = \{enc \in E_{HC} : d \in D(enc)\}$.

$$eh(sd, E_{HC}) = \frac{1}{\#sd} \sum_{d \in sd} h(E_d) \tag{4.2}$$

The difference between the expected homogeneity for a semantic decision and the homogeneity of the current set of encounters is called the *homogeneity gain* (Δh) defined in equation 4.3.

$$\Delta h(sd, E_{HC}) = eh(sd, E_{HC}) - h(E_{HC}) \tag{4.3}$$

We will only consider semantic decisions whose Δh is greater than a threshold η. The best decision will be the one with the highest Δh. If several semantic decisions in SD_1 have the same Δh then the best decision is selected in lexicographic order.

If none of the semantic decisions in SD_1 fulfills this condition, then we start the procedure once again selecting the semantic decisions in the second layer of priority of $\leq_D$ (i.e., we select the semantic decisions in $SD_2 = \{sd_i \in SD : \nexists sd_j \in SD, sd_i \neq sd_j, sd_j \notin SD_1 | sd_j \leq_D sd_i\}$. We follow the same procedure with the rest of the layers of $\leq_D$ until we find a semantic decision whose Δh is greater than the threshold η. If none of the semantic decisions in SD fulfills this condition, then the function will not be able to provide a best decision.

To solve the induction of a therapeutic sequence from a state S_1 starting from the set of encounters $\epsilon(S_1)$ we propose two algorithms: the *non-incremental determination* algorithm and the *incremental determination* algorithm. To induce the therapeutic sequence incrementally we apply the non-incremental determination only the first time, which does not have information about a previous therapeutic sequence. Then, the incremental determination is used when we already have a therapeutic sequence and we want to incorporate information about new encounters. This may lead to a different therapeutic sequence from S_1.

Both algorithms are explained in the next paragraphs.

Non-incremental determination algorithm: Algorithm 3 shows the procedure followed in the non-incremental determination of a therapeutic sequence.

This is a recursive algorithm which receives as input a set of encounters E_{HC} and a set of semantic decisions SD. We omit showing the different background knowledge

Algorithm 3: non-incremental_induction_of_TP

Input: E_{HC}: set of encounters (only the health care measures), SD : semantic decisions

Output: DT : decision tree

```
1  calculate_similarities(E_HC);
2  Create a root node for DT;
3  Update counters of DT;
4  if similar_action(E_HC) then
5      DT is a leaf labeled with best_action(E_HC);
6      Store E_HC in DT;
7  else
8      if ∀enc_i, enc_j, D(enc_i) = D(enc_j) ∨ (SD = ∅) then
9          DT is a leaf labeled with all the different actions in E_HC;
10         Store E_HC in DT;
11     else
12         sd ← best_decision(E_HC, SD);
13         if (sd=null) then
14             DT is a leaf labeled with all the different actions in E_HC;
15             Store E_HC in DT;
16         else
17             foreach d_i ∈ sd do
18                 E'_HC ← encounters enc_i in E_HC such that d_i ∈ D(enc_i);
19                 if E'_HC ≠ ∅ then
20                     Add a new branch b_i below DT labeled d_i;
21                     DT' ← non-incremental_induction_of_TP((E'_HC, SD − sd));
22                     Add the subtree DT' below b_i;
23 return DT;
```

structures and functions used in the algorithm. In the first call of the algorithm, the parameter SD is equal to the set SD of hyperedges of the semantic decisions hypergraph H_D. First of all, the algorithm calculates the similarity between actions $s(A(enc_1), A(enc_2))$ for each pair of encounters $enc_1, enc_2 \in E_{HC}$ with the procedure described in section 3.1.5 (line 1). All the similarities are calculated in this previous step avoiding to recalculate them each time they are needed during the induction process. Then it creates a root node for the DT (line 2). The next step consists in updating the counters of the DT (line 3). In case that all the encounters in E_{HC} have a *similar action* (line 4) the algorithm places the leaf with the *best action* for E_{HC} (line 5) and stores the current set of encounters E_{HC} in the DT for future uses (line 6). Otherwise, it checks if all the encounters in E_{HC} have the same decision terms or if there are no more semantic decisions left (line 8). In this case, the algorithm places a leaf labeled with all the different actions in E_{HC} (line 9) and stores E_{HC} in the DT (line 10). On the contrary, it determines the *best decision sd* (line 12). If the function is not able to find any useful semantic decision (line 13), the algorithm places a leaf labeled with all the different actions in E_{HC} (line 14) and stores E_{HC} in the DT (line 15). Otherwise, for each decision term d_i in the semantic decision sd, we determine the set of encounters $E' \subset E$ that have d_i (line 18). If the set of encounters E' is not empty, a new branch b_i is created labeled d_i (line 20). Then the algorithm is called recursively receiving as parameter the set of encounters E', and the set of remaining semantic decisions $SD - sd$, returning a DT which is connected to b_i (lines 21-22).

Notice that in lines 9 and 14 the algorithm places a leaf labeled with all the different actions in E. This does not refer to syntactically different actions but to semantically different actions. In this two cases the homogeneity of the actions in E_{HC} is lower than the threshold δ and thus similar_action(E_{HC})=false. However, some of the actions in E_{HC} may be similar enough to be considered equivalent. To solve this problem we perform an agglomerative hierarchical clustering (LVRC12b) over the set of actions in E_{HC} (see section 2.3). We use a dendrogram based on the similarity function s described in section 3.1.5 and the similarity threshold δ. Being $A(E_{HC}) = \{A_1, A_2, ..., A_n\}$ the set of actions in E_{HC}, we assign each action of $A(E_{HC})$ in a different cluster. A dendrogram can be created by successively unifying the two clusters with a higher value of *similarity*. The similarity between two clusters is calculated as the average of the similarities between each action of one cluster with all the actions in the other cluster

(average linkage clustering). The similarity threshold δ is used to cut the dendrogram obtaining a final clustering $C_\delta = \{c_1, c_2, ..., c_n\}$ where each cluster c_k contains a set of semantically equivalent actions. Finally, the leaf is labeled with one representative action of each cluster. This action is chosen with the method explained in the description of the function best_action(E_{HC}).

Incremental determination algorithm: Algorithm 4 contains the incremental algorithm to update an existing therapeutic sequence with new encounters.

Algorithm 4: incremental_induction_of_TP

Input: DT: decision tree, E_{HC}: set of encounters (only the health care measures), SD : semantic decisions

Output: DT: decision tree

1 calculate_new_similarities(E_{HC});
2 introduce_encounters(DT, E_{HC}, SD);
3 ensure_best_decision(DT, SD);
4 return DT;

The incremental algorithm receives as input the same parameters as the non-incremental algorithm plus the previous DT representing a therapeutic sequence. We omit showing the different background knowledge structures and functions used in the algorithm. In the first call of the algorithm, the parameter SD is equal to the set SD of hyperedges of the semantic decision hypergraph H. Firstly, the algorithm calculates the similarities that involve new actions included in the set of encounters E_{HC} in order to avoid repetitive calculations during the induction (line 1). Then, it basically consists of two steps. The first step, as described below, is to incorporate the new encounters into the DT by passing them down the proper branches until they reach their proper leaf (line 2). The second step, also described below, is to traverse the DT from root to leaves, restructuring it as necessary so that each internal node employs the best available decision at each moment (line 3). The procedure for the first step is detailed in algorithm 5.

It starts by updating the counters (line 1). If DT is an internal node (line 2), the new encounters are recursively passed through the branches of DT and they are introduced in the proper internal nodes (lines 3-9). Each one of these updated internal nodes is marked as stale (line 3). Notice that before applying the incremental algorithm

Algorithm 5: introduce_encounters

Input: DT: decision tree, E_{HC}: set of encounters (only the health care measures), SD : semantic decisions

```
1  Update counters of DT;
2  if DT is an internal node then
3      Mark DT as stale;
4      sd ← semantic decision used in this node;
5      foreach branch b_i below DT labeled d_i do
6          E'_HC ← encounters enc_i in E_HC such that d_i ∈ D(enc_i);
7          if E'_HC ≠ ∅ then
8              DT' ← subtree below b_i;
9              introduce_encounters(DT', E'_HC, SD − sd);
10 else
11     if similar_action(E_HC + ε(DT)) then
12         Label DT with best_action(E_HC + ε(DT));
13         Store E_HC in DT;
14     else
15         DT = non-incremental_induction_of_TP(E_HC + ε(DT), SD − sd);
```

all the nodes of the DT are not stale. In case that DT is a leaf (line 10), it checks if the encounters in DT (represented as $\epsilon(DT)$) plus the set of new encounters E_{HC} still have a *similar action* (line 11). If so, DT is labeled with the *best action* (line 12) and the new encounters are stored in DT (line 13). On the contrary, the non-incremental algorithm is called in order to obtain a subtree for $E_{HC} + \epsilon(DT)$ (line 15).

Following with the explanation of algorithm 4, at this point the new encounters have been introduced in the previous DT, updating counters and generating new subtrees if necessary. Moreover, all the existing internal nodes that have been modified by new encounters, have been marked as stale. The other step before returning the new DT is a call to a procedure that restructures the DT as necessary so that each internal node employs the best available decision at each moment (line 3). It is fully described in algorithm 6.

Algorithm 6: ensure_best_decision

Input: DT: decision tree, SD : semantic decisions
Output: DT: decision tree

```
1  if DT is an internal node ∧ DT is stale then
2      if similar_action(ϵ(DT)) then
3          DT is a leaf labeled with best_action(ϵ(DT));
4          Collect and store ϵ(DT) in DT;
5      else
6          if ∀enc_i, enc_j, D(enc_i) = D(enc_j) then
7              DT is a leaf labeled with all the different actions in ϵ(DT);
8              Collect and store ϵ(DT) in DT;
9          else
10             sd ← best_decision(ϵ(DT), SD);
11             pull_up(DT,sd);
12             Mark DT as not stale;
13             foreach successor DT′ of DT do
14                 ensure_best_decision(DT′, SD − sd);
15 return DT;
```

This is a recursive algorithm that only involves modified internal nodes. Therefore, it first checks if DT is an internal node and whether it is marked as stale (line 1). If

so, it determines if it should be a leaf with the function *similar_action* (line 2) and if it is the case, the node is assigned the best action (line 3). Otherwise, the algorithm first checks if all the encounters in $\epsilon(DT)$ have the same decision terms (line 6) and if it is the case it places a leaf labeled with all the semantically different actions in $\epsilon(DT)$ (line 7)[1]. In lines 4 and 8, as we are placing a leaf, we have to store the set of encounters in the DT. The current node was an internal node so it does not contain any set of encounters. Therefore we have to collect these encounters from the leaves of DT and store them. In the case that DT is not transformed into a leaf (line 9), we determine the *best decision* (line 10) and then we perform a *pull-up* of this semantic decision (line 11). The *pull-up* of a semantic decision consists in moving it to the top of the DT so that $pull\_up(DT, sd)$ always returns a DT with the semantic decision sd at its root. The algorithm is detailed in 7.

In the pull-up procedure, if DT is a leaf the algorithm transforms it into a subtree with the semantic decision sd in the root (lines 2-11). Otherwise, if it is an internal node with a semantic decision sd' such that $sd' \neq sd$, the algorithm can deal with 3 cases. In the general case (line 27) it makes a *pull-up* of sd for each one of the subtrees of DT (lines 28-29). This procedure is performed to fulfill the preconditions of the *transpose* operation depicted in figure 4.5, which moves the best semantic decision to the top of the DT (line 30).

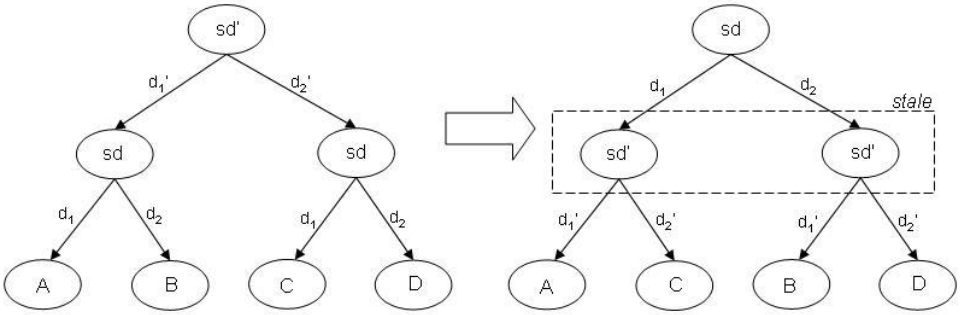


Figure 4.5: Transposing a decision tree

Notice that we have a DT with an internal node that uses the semantic decision sd' and, in the next level of the DT, we have used the *pull-up* operator in order to

[1]Here we refer to the set of encounters $\epsilon(DT)$ in the decision tree to clarify the explanation but instead of sets of encounters we use counters.

Algorithm 7: pull_up

Input: DT: decision tree, sd: semantic decision

```
1  if DT is a leaf then
2      DT is not a leaf;
3      Update counters of DT;
4      foreach d_i ∈ sd do
5          E'_HC ← encounters enc_i in ε(DT) such that d_i ∈ D(enc_i);
6          if E'_HC ≠ ∅ then
7              Add a new branch b_i below DT labeled d_i;
8              DT' ← non-incremental_induction_of_TP((E'_HC, SD − sd);
9              Add the subtree DT' below b_i;
10     Remove encounters from DT;
11 else
12     if semantic decision used in DT is not sd then
13         if DT has exactly one successor DT' which is not a leaf then
14             pull_up(DT',sd);
15             DT ← DT';
16             E'_HC ← encounters in the successors which are leaves;
17             introduce_encounters(DT, E'_HC, SD);
18         else if all the successors of DT are leaves then
19             foreach d_i ∈ sd do
20                 E'_HC ← encounters enc_i in ε(DT) such that d_i ∈ D(enc_i);
21                 if E'_HC ≠ ∅ then
22                     Add a new branch b_i below DT labeled d_i;
23                     DT' ← non-incremental_induction_of_TP((E'_HC, SD − sd);
24                     Add the subtree DT' below b_i;
25         else
26             foreach successor DT' of DT do
27                 pull_up(DT',sd);
28             transpose(DT);
```

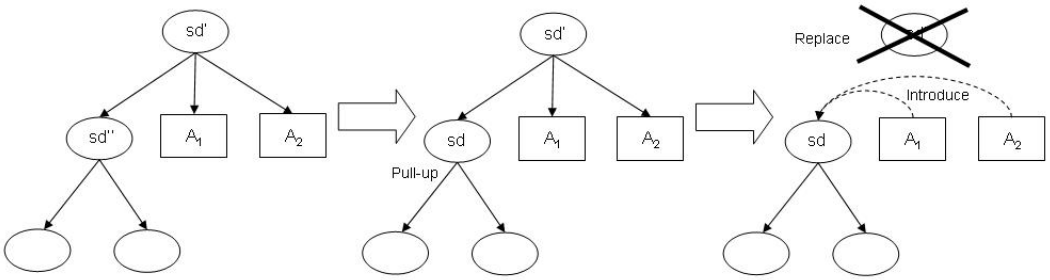
have subtrees that use the semantic decision *sd*. We do not mind about the next level of the tree which will contain internal nodes or leaves (nodes A, B, C and D in the figure). With these preconditions, we can *transpose* the DT in order to move *sd* to the root. Observe that the subtrees A, B, C and D do not need to be examined or revised. Each subtree corresponds to a set of encounters, and because the set has not changed during the *transposition*, the subtree does not need to be changed in any way. Similarly, the set of encounters corresponding to the root node of the DT does not change. Therefore, only the information in the nodes in the middle needs to be revised. This is done inexpensively, without re-examining encounters, by simply combining the information kept in their counters. The semantic decision of the nodes in the middle has not changed due to the incorporations of new encounters but because we needed to satisfy the preconditions of the *transpose* operation. Therefore, these nodes will have to be revised so they are marked as stale.

In the example used to explain the *transposition* we have supposed that both semantic decisions *sd* and sd' contained two decision terms. Usually, there will be more than two decision terms in a semantic set, but the procedure is equivalent.

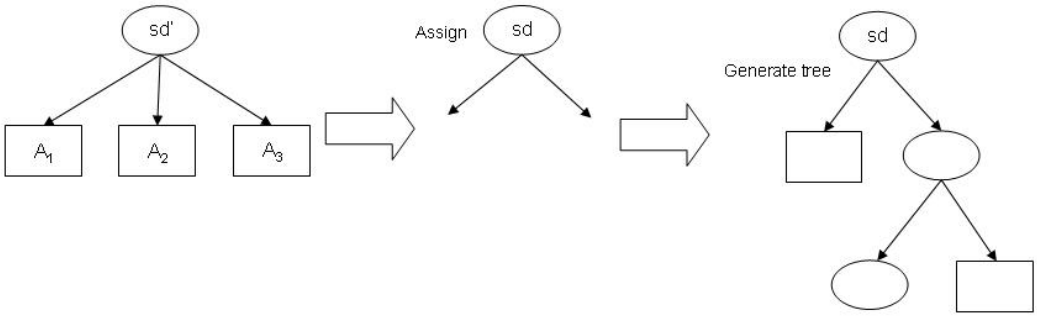
As we stated before, when performing a pull-up, we can find two exceptions to the general case (line 27) which are depicted in figure 4.6. The first case, in figure 4.6(a), is when the DT has exactly one successor which is a not a leaf (line 14). Here we move *sd* to the top of the successor which is not a leaf using a pull-up (line 15). Then we replace the current DT by this successor (line 16) and introduce the encounters of the successor leaves (lines 17-18). The second exception, in figure 4.6(b), is when all the successors of DT are leaves (line 19). In this case we assign the best decision to DT and generate the branches below it using the non-incremental algorithm (lines 20-26).

Continuing with algorithm 6, after the pull-up procedure, we already know that the node uses the correct semantic decision so we can mark it as not stale (line 12). Then we recursively call the same algorithm for each one of the successors of DT (lines 13-14). With this procedure, the modified subtrees of DT are revised so that we finally ensure the best decision in each node of the decision tree.

In the following we will illustrate an example of how the presented algorithm updates a therapeutic sequence when introducing new encounters. Consider the DT in figure 4.7 which represents a simple therapeutic sequence that could be obtained during the generation of a SDA diagram for the treatment of hypertension.



(a) DT has exactly one successor which is not a leaf



(b) All the successors of DT are leaves

Figure 4.6: Two exceptions to the general case when performing a pull-up

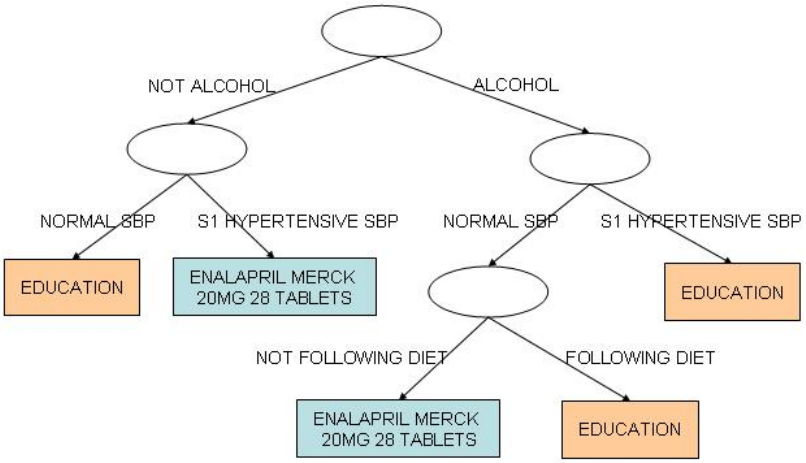


Figure 4.7: Example of updating a therapeutic sequence (1)

As soon as new encounters arrive, we have to introduce them into the DT and then ensure the best decision for each node. In figure 4.8 some encounters have been introduced which have passed through the branches *ALCOHOL* and *S1 HYPERTENSIVE SBP*. The nodes which have incorporated new encounters are marked as stale (painted in gray). Some of these new encounters do not contain the action *EDUCATION* proposed by the DT, so the corresponding leaf has been replaced by a subtree.

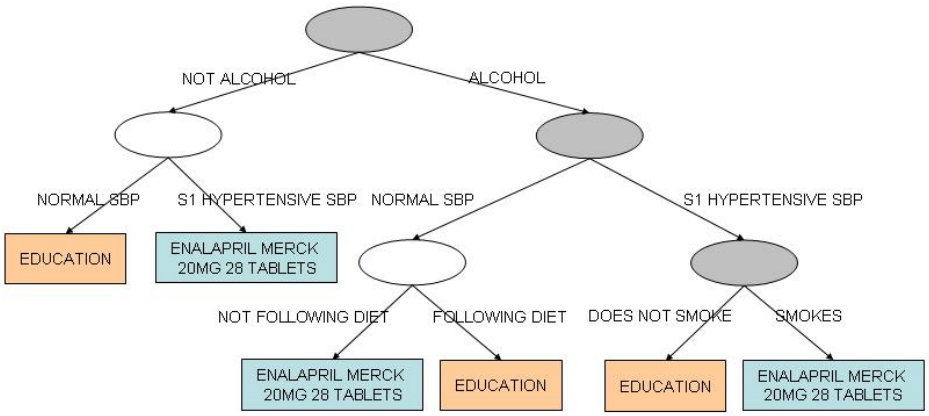


Figure 4.8: Example of updating a therapeutic sequence (2)

Once the encounters have been introduced into the DT, we ensure the best decision for each node. Now we suppose that the best semantic decision for the root node is the one related to SBP. This node currently separates patients that take alcohol from those who do not, so we have to perform a pull-up. As we have the desired decision in both successors of the root node we simply transpose the DT (see figure 4.9). We have to mark as stale the internal nodes during the transposition and remove the stale mark from the root node.

Finally, we ensure the best decision for each remaining stale node. We suppose that the node at the left that asks for alcohol already contains the best decision and the one at the right should ask for smoking habits. In the latter case, we make a pull-up following the first exception (the DT has exactly one successor which is not a leaf) and obtain the final DT in figure 4.10.

The incremental determination of therapeutic sequences includes several features that allow dealing more efficiently with the constant arrival of data of new encounters. For each new incremental determination of a therapeutic sequence we do not have to induce the whole DT from scratch. The new encounters are introduced through the

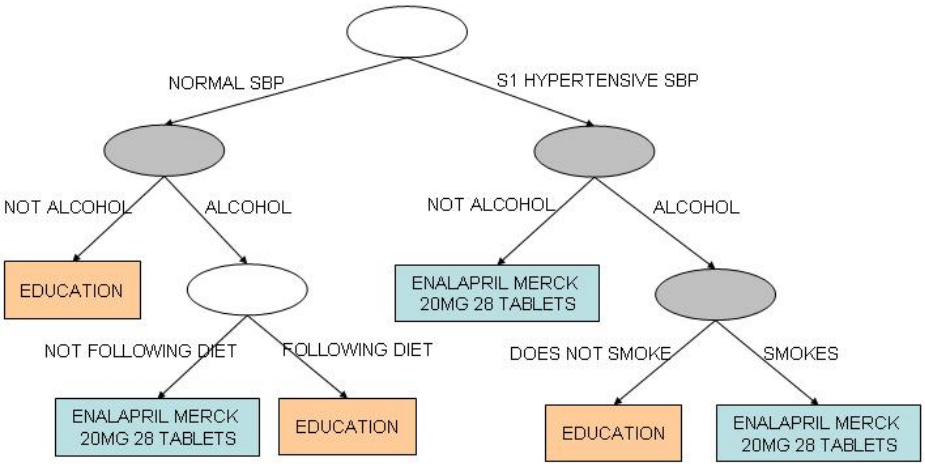


Figure 4.9: Example of updating a therapeutic sequence (3)

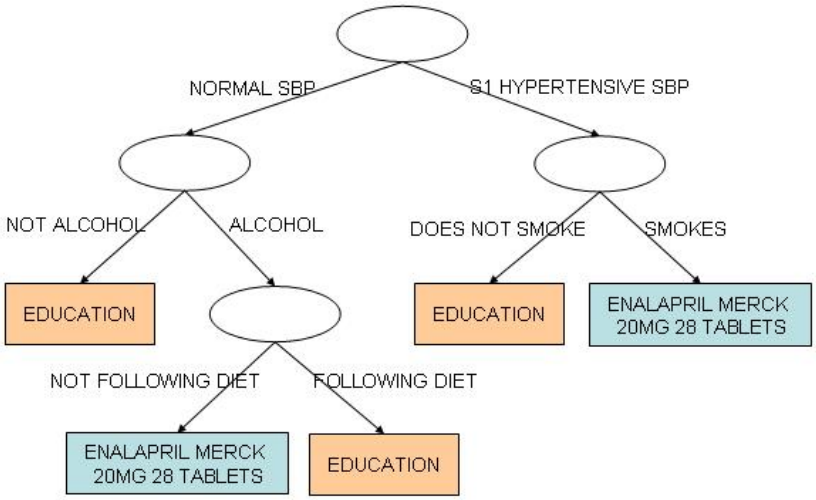


Figure 4.10: Example of updating a therapeutic sequence (4)

existing DT creating new branches if necessary. Finally only the nodes of the DT which have been affected by the new encounters are revised and restructured so that the best decision is made at each node. Therefore, the therapeutic sequence only depends on the encounters used, without regard to the sequence in which those encounters were presented.

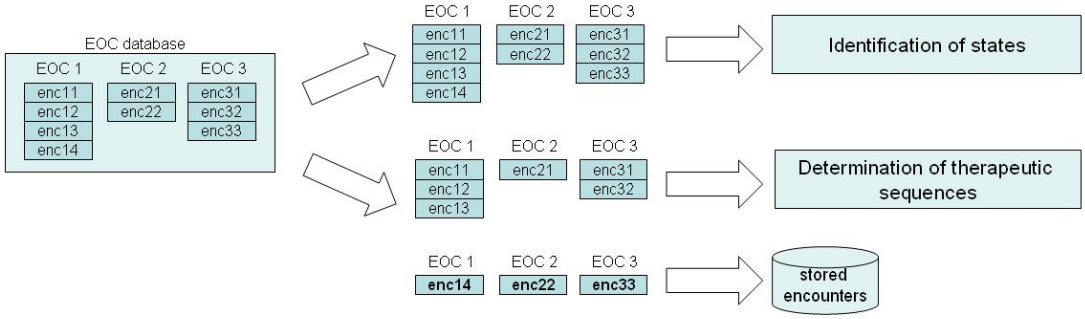
Another advantage of the algorithm is that instead of storing sets of encounters in each internal node of the DT, it uses counters that represent essentially the same information but in a more efficient way. With this representation the amount of memory used by the DTs does not grow with the number of encounters and it still keeps all the information needed to induce the therapeutic sequences.

4.3 Integration of the procedures to generate SDA diagrams

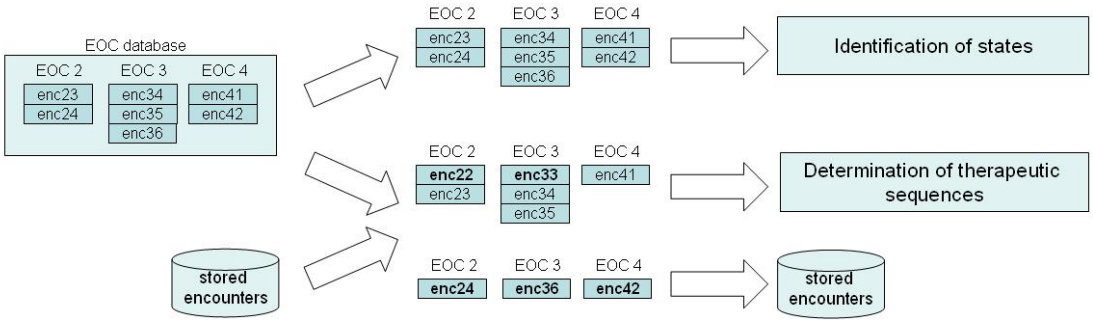
Once the procedures of identifying the states and determining the therapeutic sequences have been detailed, in this section we will explain how these two procedures are integrated in order to generate SDA diagrams.

We start considering that the identification of states is based on the data contained in each one of the encounters. Observe also that therapeutic sequences start from a certain state and that this state and treatment leads to another state. This next state is not included in the therapeutic sequence itself but it is necessary in the integration step in order to make the proper connections. To generate therapeutic sequences we use data contained in each one of the encounters, but to connect these therapeutic sequences we need to know the state of each following encounter. Therefore, for each one of the episodes of care in the EOC database we will have to discard the last encounter because we still do not know the state of its following encounter. These last encounters will be used in the identification of states but will be ignored in the determination of therapeutic sequences. Nevertheless, these encounters will not be lost but they will be stored together with the identification number of their corresponding episode of care. The next time that we update the SDA diagram, the stored encounters are added at the top of their respective episode of care. This procedure is depicted with an example in figure 4.11. The first time that we generate a SDA diagram we do not use the last encounters of each EOC ($enc_{14}, enc_{22}, enc_{33}$) in the determination of therapeutic

sequences but we store them for future uses (see figure 4.11(a)). In a next incremental generation of the SDA diagram we recover the stored encounters of the involved EOCs and store the last encounters of each current EOC (see figure 4.11(b)). Notice that enc_{14} is not recovered this time because there are no new encounters of EOC 1 in the database.



(a) Example of storage of encounters in the initial generation of the SDA diagram



(b) Example of storage and recovery of encounters in a next incremental generation of the SDA diagram

Figure 4.11: Storage and recovery of encounters in the generation of the SDA diagram

In order to generate SDA diagrams integrating the procedures explained in the previous sections we follow these steps:

1. Identification of states and transformation to the SDA model
2. Determination of therapeutic sequences from the identified states and transformation to the SDA model
3. Connection of SDA actions with the corresponding following SDA states

4. Selection of a suitable level of abstraction for SDA action terms

5. Unification of SDA actions

The first step consists in applying the procedure of identification of states described in section 4.1 using all the encounters in the EOC database. The first time that we generate the SDA diagram we will identify these states from scratch while the next times, the set of states of an existing SDA diagram will be modified with the incremental algorithm. The resulting set of states is easily transformed to the SDA model creating a SDA state for each one of the states identified that contains the corresponding set of SDA terms. Notice that for each incremental generation of the SDA diagram, the set of identified states may change and thus some SDA states may be removed from the SDA diagram and some new SDA states may appear.

In the second step, a therapeutic sequence (decision tree) is determined for each of the identified states in step 1. As we explained before in this section, the last encounter of each episode of care in the EOC database is ignored and stored for the future while the rest of encounters are used to apply the procedure in section 4.2. For the non-identified states we do not have to induce the DT representing its therapeutic sequence because it will not be included in the SDA diagram. However, the non-identified states also represent sets of encounters which have to be stored because they may become identified states in the future. When we update the SDA diagram with new encounters, some non-identified states may be identified and we must consider all the encounters that have been in this state in order to induce a correct therapeutic sequence. Therefore, we will store the set of encounters for each non-identified state in order to be used in the future. Similarly, a state which was identified in a previous generation may not be identified in the current one. In this situation we created a therapeutic sequence for this state which now will not appear in the SDA diagram. This therapeutic sequence must be stored so that, in the future, it can be updated with new encounters. In general, we will deal with 14 different situations which are summarized in table 4.3.

The first row corresponds to the case of the first time we generate the SDA diagram where neither encounters nor DTs have been stored. If the state is identified we generate and store the DT which will appear in the SDA diagram. Otherwise, we store the new encounters for future uses. For the first generation, the cases where there are no new encounters are impossible. In the second row, for the current state, some encounters

Table 4.3: Summary of operations with DTs and the stored encounters for each possible situation

	Identified		**Not identified**	
Stored	**New encounters**	**No new encounters**	**New encounters**	**No new encounters**
Nothing	Generate/store DT	-	Store encounters	-
Encounters	Merge encounters, reset old encounters and generate/store DT	Reset old encounters and generate/store DT	Merge/store encounters	No operation No operation No operation
DT	Update/store DT	No operation	Store encounters	No operation
Encounters + **DT**	Merge encounters, reset old encounters and update/store DT	Reset old encounters and update/store DT	Merge/store encounters	No operation No operation No operation

where stored in a previous generation. If the state is identified, a DT must be created using the stored encounters and thus they can be removed. Moreover, if there are new encounters they must be also considered so we merge old and new encounters before generating the DT. If the state is not identified there is no need to generate a DT. We merge old and new encounters if necessary. Following the third row, we may deal with the situation where a DT was stored previously for the current state. If this state is identified we must update the DT with the new encounters and if it is not identified the new encounters are stored. If there are no new encounters, no operation is needed. Finally, in the forth row, we have the case where we have both old encounters and an old DT. In this case, the operations to be followed are the same that when we only have old encounters stored, but if the state is identified, we update the old DT instead of generating it from scratch.

At this point we have a set of DTs which represent therapeutic sequences that start from each of the identified states. These therapeutic sequences are then transformed to the SDA model by replacing the internal nodes of the tree by SDA decisions and the leaves by SDA actions. Each branch leading from an internal node is labeled with a decision term, and they are transformed to decision connectors containing this term and pointing to the next internal node (the branches labeled 'otherwise' are transformed into otherwise connectors). The leaves are labeled with an action containing a set of action terms, so a different SDA action is created for each leaf, containing its action terms. Finally, each therapeutic sequence is connected with its respective identified state. Therefore, in the SDA diagram we place a plain connector from each SDA state created in step 1 pointing to the SDA decision in the root of the corresponding

therapeutic sequence.

In the third step we connect each therapeutic sequence to the following next states. In order to do this we must have stored, for every encounter of each identified state, which is the state of the following encounter. Therefore, for each SDA action at the end of a therapeutic sequence we will have a set of following states. In the SDA diagram, we create a plain connector leading from a SDA action to each one of the corresponding SDA states (type-2 non-determinism).

After the third step, the SDA diagram is already finished. However, two modifications are made that improve the visualization of the SDA diagram. The forth step consists in selecting a suitable level of abstraction for the SDA action terms. The threshold δ defined in the process of calculating if a set of encounters have a similar action in section 4.2.3 is related to the level of abstraction of the actions in the diagram. If δ has a low value, we want our final diagram to consider as equivalent two actions with a low similarity. Thus if, for example, $\delta = 0.3$ two different diuretics like hydrochlorothiazide and indapamide will be considered equivalent in the diagram. Therefore, if we specify such a low δ, we are interested in knowing that the treatment contains a diuretic rather than knowing if it should be hydrochlorothiazide or indapamide. Otherwise, with a higher value of δ, these two diuretics will not be considered equivalent and they will lead to two different actions in the diagram. In this case, we are interested in knowing exactly which kind of diuretics are needed. Therefore, with δ we are actually defining a level of abstraction for the terminology in the SDA actions. As a consequence, for each one of the action terms in the SDA actions we will go up through its predecessors in the action terms hierarchy $\mathcal{H}_A$ and we will choose the last concept whose successors are considered equivalent according to δ. Following the previous example, with $\delta = 0.3$, we may find the action term *HIDROSALURETIL 50MG 20 TABLETS* in one of the SDA actions. Going up one level we find out that its active principle is *C03AA03 hydrochlorothiazide* whose successors have a similarity of at least 0.7. The same happens with the next level which is *C03AA Thiazides, plain*. We keep going up to *C03A LOW-CEILING DIURETICS, THIAZIDES* whose successors have a similarity of 0.5. The successors of the next concept *C03 DIURETICS* have a similarity of 0.3, so they are still considered equivalent. This would be the level of abstraction selected because the next one, *C CARDIOVASCULAR SYSTEM*, has successors with similarity equal to 0.0 ($< \delta$). So we choose the concept *C03 DIURETICS* to replace the original action

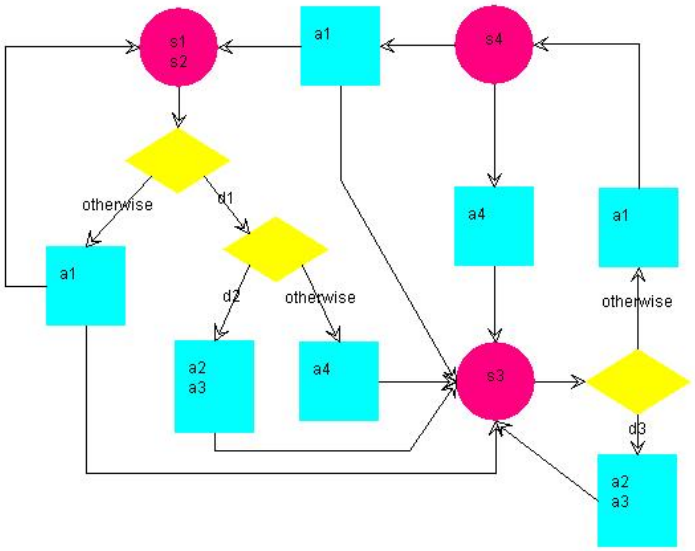
term *HIDROSALURETIL 50MG 20 TABLETS*. With this procedure we guarantee a proper terminology for actions in the final SDA diagram.

The other modification done in the fifth step aims at reducing the number of SDA actions in order to simplify the diagram. Our procedure to generate SDA diagrams creates a different DT for each state. Each one of these DTs has several SDA actions as leaves. A concrete SDA action may appear several times in the diagram in different therapeutic sequences. It is also possible that some of these equal SDA actions lead to the same states. In these cases, we can unify these SDA actions because they represent the same treatment followed by the same expected transition to a next state. In the fifth step, we detect sets of equal SDA actions that are also connected to the same next states. We unify these SDA actions by removing all of them except one and redirecting each connector that pointed to the removed SDA actions, to the SDA action that remains in the diagram as it is depicted in figure 4.12. In figure 4.12(a) there is a SDA diagram with 7 actions. The action containing the action terms $\{a_2, a_3\}$ and pointing to the state $\{s_3\}$ is repeated twice and thus they can be unified. The same happens with action $\{a_1\}$ pointing to states $\{s_1, s_2\}$ and $\{s_3\}$ and with action $\{a_4\}$ pointing to state $\{s_3\}$. The resulting SDA diagram after unifying SDA actions is depicted in figure 4.12(b). Notice that there are still two actions containing $\{a_1\}$ but they could not be unified because they are pointing to different states.

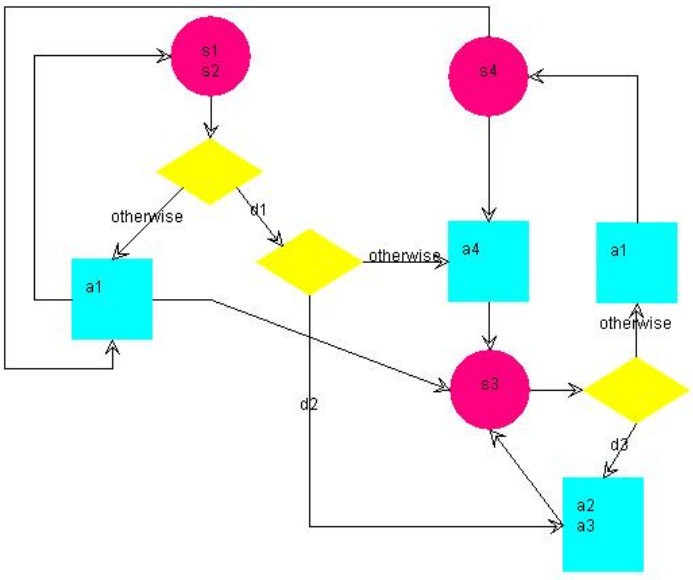
4.4 Summary of the incremental generation of SDA diagrams with background knowledge

This chapter has introduced two procedures to automatically induce two basic elements of medical procedural knowledge structures, concretely SDA diagrams, which are health care states (see definition 3.1.1) and therapeutic sequences (see definition 3.1.5). Both procedures automatically generate these medical structures from an EOC database which contains episodes of care, encounters, etc. of patients treated for a certain pathology. They are also based on background knowledge which is not explicit in the EOC database. Finally, both procedures are able to work in an incremental way in order to deal with the constant arrival of new medical data.

Table 4.4 includes a list of all the parameters that have been used during the explanation of both procedures:



(a) SDA diagram with 7 actions before unification



(b) SDA diagram with 4 actions after unification

Figure 4.12: Unification of SDA actions

Table 4.4: Parameters of the procedures of identification of states and determination of therapeutic sequences

Parameter	Explanation
δ	threshold to determine whether two or more action terms or actions are equivalent or not
α_e	weight given to the epidemiological view of the quality of a state
α_t	weight given to the therapeutic view of the quality of a state
α_p	weight given to the preferential view of the quality of a state
$\min_S$	minimum number of terms per state
$\max_S$	maximum number of terms per state
η	minimum homogeneity gain for semantic decisions to be chosen as best decisions

A method to integrate both procedures has been presented at the end of the chapter in order to incrementally generate whole SDA diagrams using background knowledge.

5

Tests and results

There are three main aspects that we want to verify about the proposed methodology which are:

1. The good performance at the levels of the use of background knowledge and incrementality.

2. The adherence of the obtained SDA diagrams to the databases used to generate them.

3. The medical correctness and comprehensibility of the obtained SDA diagrams.

With regard to the first aspect, we have designed some tests to determine whether the use of background knowledge implies an increase of temporal cost or not, and whether the methodology fulfills the desirable goals of incrementality.

The second aspect is tested calculating the adherence of the SDA diagrams generated for Hypertension (HT), Diabetes Mellitus (DM) and the comorbidity of both diseases to the source databases containing the details of treatment for real primary care patients assisted in SAGESSA. For the case of HT, we also generate a SDA diagram using the incremental approach during a period of time and analyze the adherence to the database during its evolution.

Regarding the third aspect, we have decided to generate definitive SDA diagrams for each one of the previous pathologies in order to test them at a medical level. Moreover, we compare these diagrams with those generate with the knowledge-free approach

in (BRLV12) from a medical point of view. Finally, we follow the evolution of the SDA diagram of HT during a period of time and analyze it.

All these tests have been done considering the background knowledge presented in section 3.3 in order to guarantee medically correct results. Both the tests and the results obtained are detailed in next sections.

Regarding the parameters (see table 4.4), we have performed several tests and consultations with the health care professionals in order to determine their value. The parameters related to the quality of a state have been assigned $\alpha_e = 0.0$, $\alpha_t = 0.3$ and $\alpha_p = 0.7$ for all the tests because the health care professionals were interested in giving a great weight to their preferences and a lower weight to the therapeutic view. With regard to the constraints about the number of terms within a state, the health care professionals decided that for all these tests it was not necessary to specify a minimum number of terms ($\min_S = 0$). The maximum number of terms $\max_S$ has been always fixed ($\max_S = 2$ for HT and HT+DM, and $\max_S = 3$ for DM). The value for parameter η has been determined after generating the same therapeutic sequences changing the values of η and showing the results to the health care professionals. The experience suggests that $\eta = 0.1$ is the best choice, so this value has been fixed for all the tests. Finally, giving different values to the parameter δ is very interesting because it let us change the level of abstraction of the terminology of the actions in the SDA diagram. Therefore, during the following tests, we have tried different values for δ as it is clearly detailed.

The proposed methodology has been implemented and integrated into an existing software called SDA Lab which is used to manually develop, manage and execute SDA diagrams.

5.1 Integration in SDA Lab

SDA Lab (LV07) is a software platform created in 2007 with the main purpose to manually develop, manage and execute SDA diagrams. Figure 5.1 contains a screen shot of SDA Lab v1.4. In v1.4, it also includes a tool to automatically generate SDA diagrams from data using the approach in (BRLV12).

We have evolved SDA Lab to v1.5, replacing the previous method to generate SDA diagrams by the knowledge-based incremental methodology presented in this thesis.

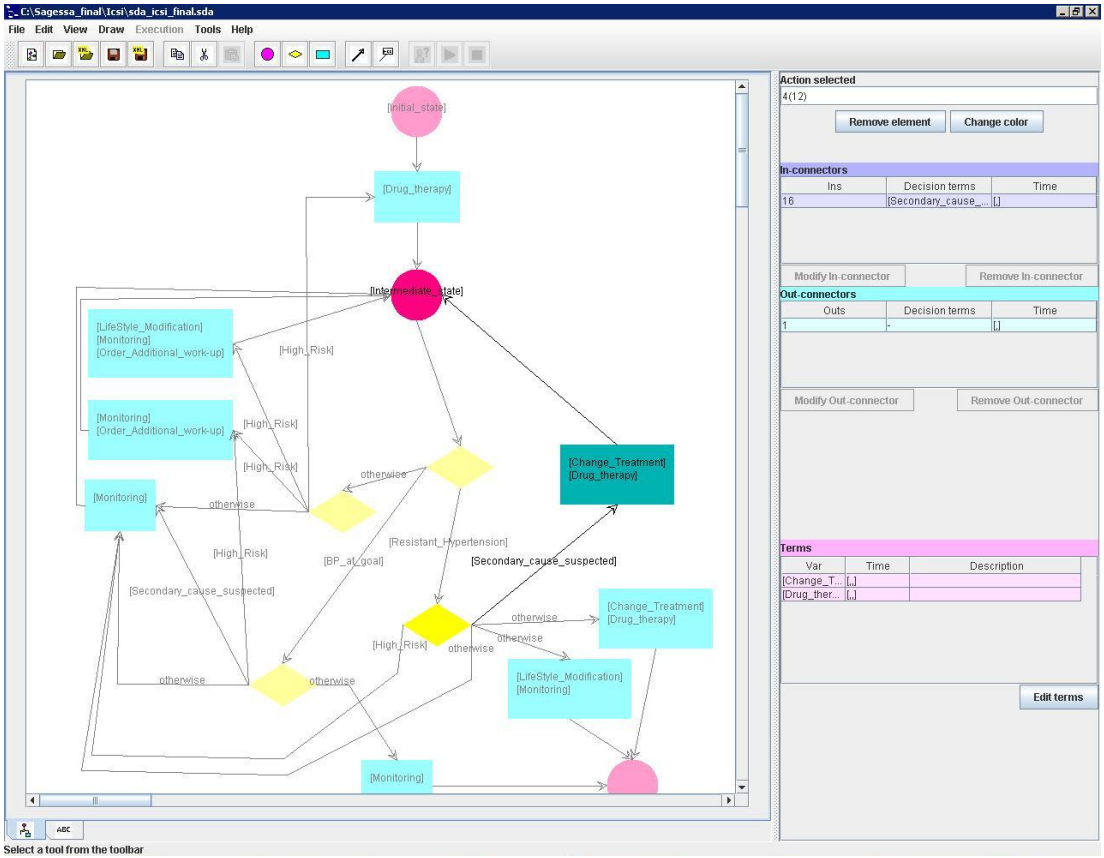


Figure 5.1: Developing a SDA diagram with SDA Lab

Concretely, SDA Lab v1.5 has the following new features:

- Incremental learning of SDA diagrams
- Incremental incorporation of encounters into a SDA diagram one-by-one
- Knowledge-based learning of SDA diagrams
- Management of background knowledge (graphs, hypergraphs, partial orders and concept hierarchies)
- Manual or automatic (from a file) creation of background knowledge structures
- Execution of SDA diagrams
- Quality assessment of SDA diagrams

When we decide to generate a new SDA diagram we can introduce or modify all the background knowledge regarding states, decisions and actions. For example, figure 5.2 depicts the window to select semantic decisions, to decide their priority and to choose a minimum level of homogeneity (the similarity threshold δ introduced in section 4.2.3).

All the tests presented in this chapter have been performed with SDA Lab v1.5.

5.2 Performance tests of background knowledge

The use of background knowledge in the generation of SDA diagrams is essential to guarantee medically correct results as we will show in future sections. However, before analyzing these benefits we have to make some performance tests in order to determine whether considering background knowledge of the medical domain implies an excessive increase of temporal cost of the knowledge-based solution presented in the thesis or not with respect to previous knowledge-free solutions (BRLV12).

In this section we compare our approach with the one proposed in (BRLV12) which generates SDA diagrams with a different methodology that does not consider the background knowledge of the domain. We have generated four SDA diagrams with 100, 200, 400 and 800 encounters with patients of HT in SAGESSA during the year 2009, using the algorithm in (BRLV12) and with our knowledge-based approach. The results obtained are shown in figure 5.3 where performance is measured in terms of the execution

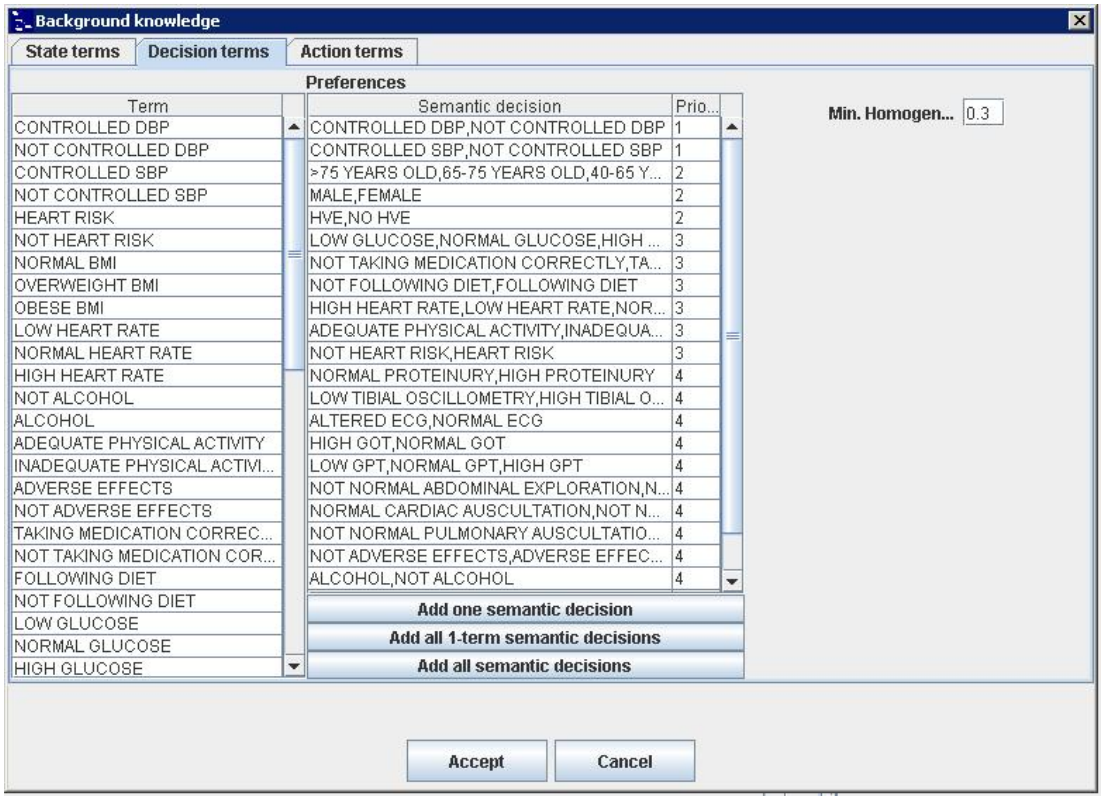


Figure 5.2: Introducing the background knowledge related to decisions with SDA Lab v1.5

time expressed in seconds (vertical axis) and conditioned to the number of encounters (horizontal axis).

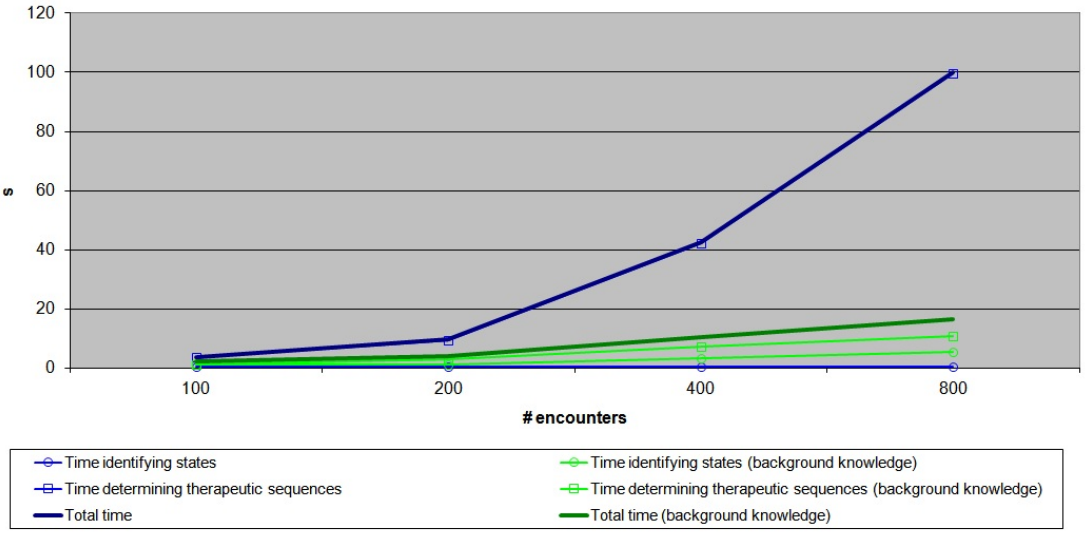


Figure 5.3: Comparison of time cost with and without background knowledge

The blue lines represent the results of the induction with the algorithm in (BRLV12) and the green lines represent the results of our background knowledge based approach. The simple method used to identify states without background knowledge in (BRLV12) implies a low temporal cost of less than 0.5 s whereas our approach, which considers several constraints between terms, partial orders, etc., has a greater cost that grows to 5.5 s (for 800 encounters). Contrarily, the determination of therapeutic sequences is favorable to our approach which grows linearly while the method in (BRLV12) clearly does not. A deeper analysis of the causes of this improvement concludes that in our knowledge-based approach the calculation of similarities between actions to detect equivalences and the use of semantic decisions drastically reduces the size of our decision trees. Moreover, the number of states identified is lower in our approach, so less decision trees are induced. This implies a small average increment of 3 s for each test until a maximum of 10.9 s, for 800 encounters. The approach to determine therapeutic sequences proposed in (BRLV12) shows a non-linear trend that reaches the duration of 100 s in the last test. This great difference between both approaches can also be reinforced by the fact that our approach reduces the number of steps used to determine therapeutic sequences,

from three (detecting actions, determining evolutions and determining actions) to one (determining therapeutic sequences).

The results obtained show evidence that the use of background knowledge not only does not imply an increase of temporal cost but it severely reduces the time spent determining therapeutic sequences and, therefore, the total duration of the process (1.6 s for 100 encounters, 5.8 s for 200, 32.2 s for 400 and 83.8 s for 800).

5.3 Performance tests of incrementality

The use of the incremental approach has to be empirically justified. In section 2.6 we introduced three desirable goals which usually motivate the use of incremental learning algorithms. These are:

1. Cost reduction: The incremental cost of updating the current hypothesis with a new instance should be much lower than the cost of building a new hypothesis from scratch. It is not necessary however that the sum of the incremental costs be less than the execution on the complete database.

2. Independence from the size: The update cost should have a high degree of independence to the number of training instances on which the decision mechanism is based.

3. Independence from the order: The hypothesis produced by the incremental algorithm should depend only on the set of instances that has been used, without regard to the sequence in which these instances were presented.

We have performed several tests to determine whether these goals are achieved or not. These tests are presented in the following sections together with the results obtained. For all of these tests we have used the data of patients of HT in SAGESSA during the year 2009.

5.3.1 Cost reduction

A desirable goal of our incremental algorithm is that the cost of incorporating a new encounter to a previously built SDA diagram should be much lower than the cost of inducing the new SDA diagram from scratch. However, it is not necessary that the sum

of the different incremental costs be less than the execution on the complete database of encounters.

In order to check whether this goal is achieved or not, we have performed the following test. Initially we induce a SDA diagram using 200 encounters. Then we incrementally incorporate a new encounter to the diagram and we calculate the time spent with this updating. We repeat 10 times this incremental procedure adding a new encounter to each resulting SDA diagram. Finally, we generate the same SDA diagrams with the non-incremental procedure (one induced with 201 encounters, one with 202 encounters, etc.) and we calculate the duration of the procedure. With this test we are able to compare the time spent updating a SDA diagram with several new encounters, with the time spent generating the same SDA diagrams from scratch. Figure 5.4 shows the results of the test.

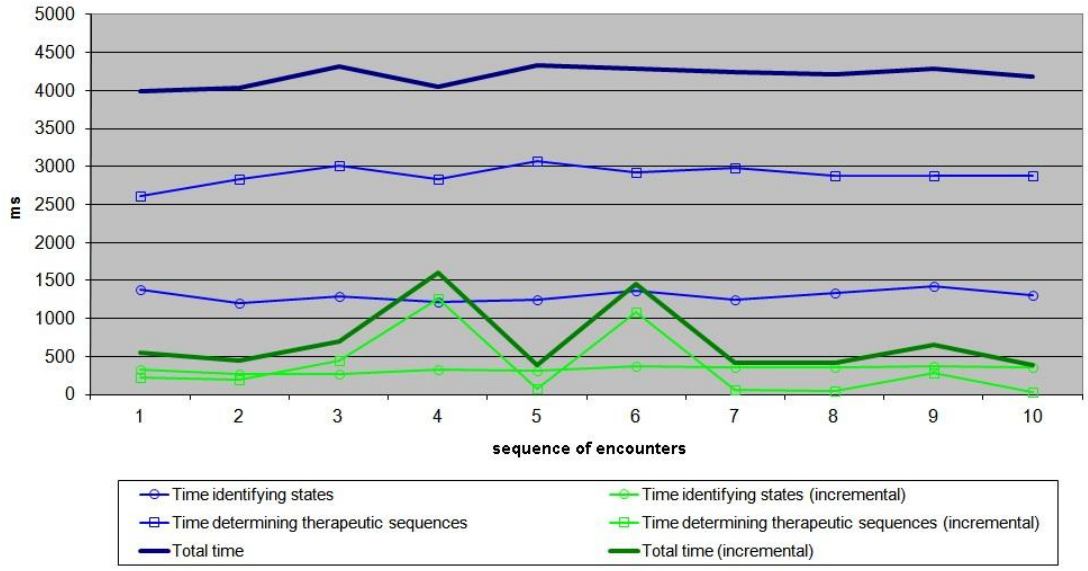


Figure 5.4: Comparison of time cost between the incremental and the non-incremental approach

The blue lines represent the results of the non-incremental induction and the green lines represent the results of the incremental induction. The durations of the identification of states are very constant in both cases, with an average of 1301.6 ms for the non-incremental approach, and 332.8 ms for the incremental approach. The time spent determining therapeutic sequences is constant in the non-incremental approach

and highly variable in the incremental approach. This is caused by the fact that the number of decision trees to be revised will not always be the same and therefore each updating will require a different number of operations of pull-up, transposition, etc. In the case of determining the therapeutic sequences the average durations are 2891.3 ms for the non-incremental approach, and 368.6 ms for the incremental one. A particular step that increases the time cost in the non-incremental approach is the calculation of similarities between actions which is performed in the beginning in order to avoid recalculations. This procedure takes about 1694.2 ms in the non-incremental approach because the similarities between all the pairs of actions have to be calculated. In the incremental approach only the similarities with new actions (if there are any) have to be calculated so the time spent is reduced to an average 41.5 ms. If we only consider the time spent generating decision trees (avoiding the time used to calculate similarities between actions) we observe that in the incremental approach, the steps that require more modifications of the existing decision trees (4,6) have a cost almost as high as the non-incremental approach, while the steps that do not require any modification of existing decision trees (5,7,8,10) have a cost equal to 0.

The total temporal cost of updating an existing SDA diagram with a new encounter is averaged 3491.5 ms lower than generating a new SDA diagram from scratch, so we can conclude that the first desirable goal is achieved.

If we update an existing SDA diagram with more than one encounter at the same step, the time spent will increase because more decision trees will have to be revised. Now we want to determine whether it is still worth or not updating an existing SDA diagram rather than generating it from scratch if the updating is performed at the end of the day. We perform another test to check this. As we did in the previous test, we induce a SDA diagram with 200 encounters. Now we incrementally update it with the set of encounters that have taken place during one day. We calculate the time spent with this updating and repeat the procedure 10 times corresponding to 10 consecutive days. We compare the duration of this procedure with the duration of generating the same SDA diagrams from scratch. In this case, the incremental approach still obtains better results of time cost. Concretely, the incremental approach is averaged 1295.2 ms faster.

Repeating the previous test but updating the SDA diagram every two days, the difference grows to 1475.6 ms. Finally, when making the test updating the SDA diagram

every three days, the incremental approach increases its costs and takes 129.6 ms more than the non-incremental approach. Therefore, the results suggest that, in the case of SAGESSA or other health care centers with a similar rate of patients per day, for SDA diagrams based on about 200 encounters, if we generate them every two days or less, the approach with lower costs is the incremental one. If we generate the SDA diagram every three days or more, then the non-incremental approach is preferred.

However, the recommended practice is always to update the SDA diagram as soon as a new encounter arrives because the temporal cost will be the lowest and the SDA diagram used as a support to make decisions will always be up to date.

5.3.2 Independence from the size

Another desirable goal of the incremental algorithm is that the cost of updating the SDA diagram with new data should have a high degree of independence to the number of encounters that have been used to generate the initial SDA diagram.

We have checked this goal by performing the following test. We generate a SDA diagram with one encounter and then we keep updating it with one new encounter until 200 encounters are incorporated. Figure 5.5 shows a linear graph with the results of the test related to the cost of identifying states.

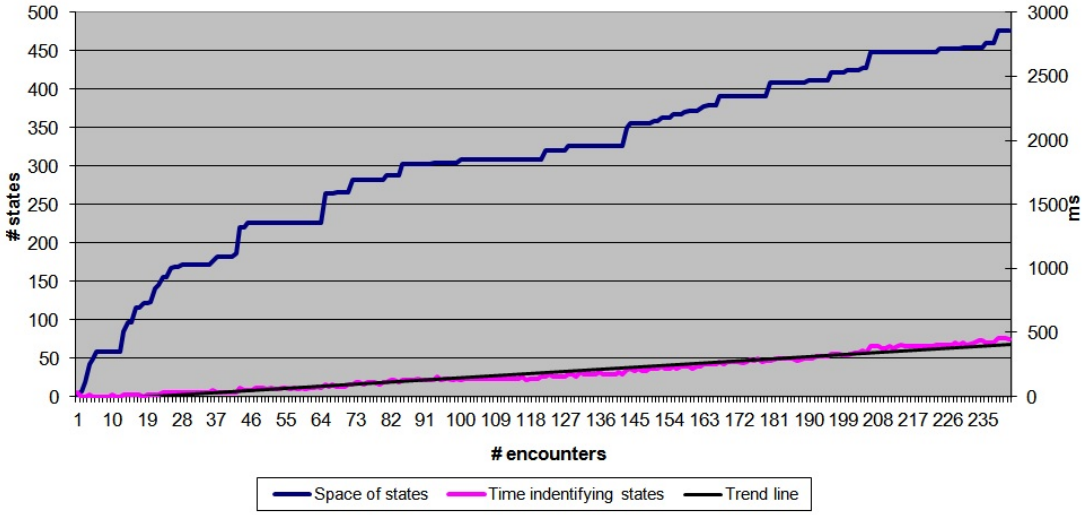


Figure 5.5: Evolution of the cost of identifying states when incorporating the first 200 encounters to a SDA diagram

The pink line shows the evolution of the time spent in milliseconds (right vertical axis) to identify states when incorporating the first 200 encounters to a SDA diagram, and the blue line represents the size of the space of states (left vertical axis) which increases with the incorporation of new encounters. The graph also includes a trend line of the evolution of the update cost. The results show that the update cost increases linearly about 1.8 ms for each new encounter incorporated. A linear trend with such slope suggests that the update cost of identifying states has a light dependence on the number of encounters used.

On the other hand, figure 5.6 shows another linear graph with the results of the test related to the cost of determining therapeutic sequences.

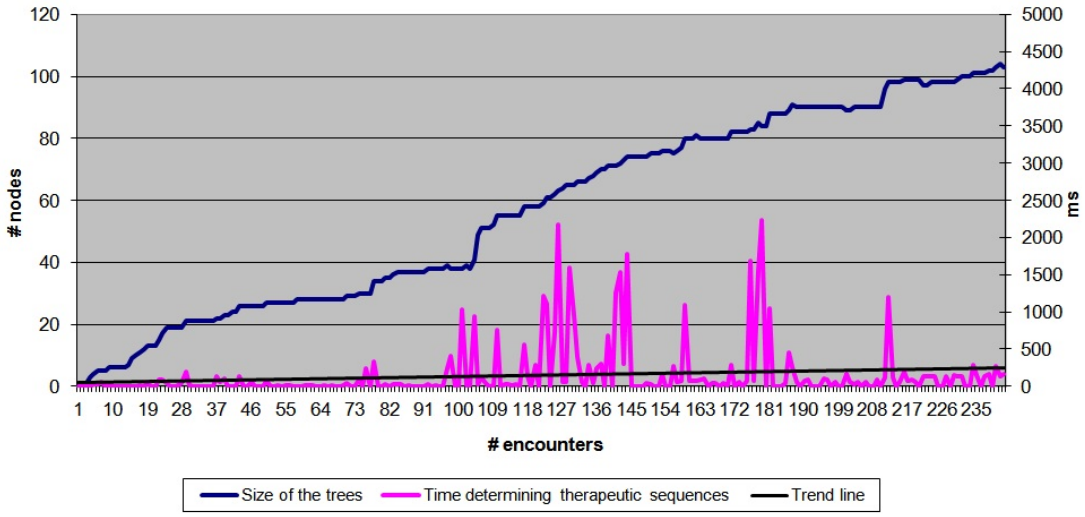


Figure 5.6: Evolution of the cost of determining therapeutic sequences when incorporating the first 200 encounters to a SDA diagram

In this case, the pink line shows the evolution of the time spent in milliseconds (right vertical axis) to determine therapeutic sequences when incorporating the first 200 encounters to a SDA diagram. We compare this evolution with the blue line which represents the total number of nodes (left vertical axis) of all the decision trees which gives us an idea of the current size of the problem of determining therapeutic sequences. Notice that the temporal cost of each updating is very variable because it may affect a different number of decision trees and also a different number of operations such as pull-up, transposition, etc. The results show that the cost increases linearly about 0.8 ms for each new encounter incorporated. Therefore, when determining therapeutic

sequences the dependence on the number of encounters used is even lower than when identifying states.

We can confirm that we achieve this desirable goal because the degree of independence to the number of encounters used is very high according to the results (an increase of 1.8 ms and 0.8 ms for each additional encounter when identifying states and determining therapeutic sequences respectively).

5.3.3 Independence from the order

The last desirable goal is that the SDA diagram generated by the incremental algorithm should depend only on the set of encounters that has been used, without regard to the sequence in which those encounters were presented.

We have checked this property with the following test. We generate a SDA diagram with 200 encounters incrementally incorporated one by one. Then we randomly shuffle these encounters and we generate once again the SDA diagram. This procedure has been done 5 times obtaining 5 SDA diagrams. In all the cases, the diagrams generated were exactly the same, concluding that the desired goal is achieved.

5.4 Database adherence tests

Once the performance has been checked, we want to determine the adherence of the SDA diagrams to the hospital database that has been used to generated them in order to conclude whether the diagrams are able to correctly represent the treatments in the database or not. Moreover, we want to generate a SDA diagram for HT using the incremental approach during year 2009 and analyze the adherence to the database during its evolution.

In section 3.1.5 we proposed a method to calculate the similarity between clinical actions. This method can be used to compare the medical treatment given to a patient in an encounter with the treatment that a SDA diagram proposes to this patient, obtaining a value of similarity between both treatments. In general, we can obtain the average similarity between the treatments in an EOC database and a certain SDA diagram.

5.4.1 Database adherence for each pathology

We have generated a SDA diagram for each of the pathologies: HT, DM and the comorbidity of both diseases, with different values of the parameter δ (see section 4.2.3), which determines the level of granularity of the actions in the diagram. If $\delta \approx 0$ then the SDA diagram will be more generic and smaller, otherwise if $\delta \approx 1$ it will be more concrete but bigger.

Table 5.1 contains the results obtained for different values of δ. Observe that with $\delta = 1$ (i.e., when no abstraction is made in medical actions registered in the database) we always obtain an average similarity of 1. The SDA diagram exactly reflects all the concrete treatments in the database. However, for extremely low values of δ, which imply a high level of abstraction in the actions of the SDA diagram, we still obtain an acceptable average similarity of 0.76, 0.68 and 0.76 respectively, for each pathology. A reduction of 0.7 in δ only implies an average reduction of 0.27 in average similarity. Moreover, the growth on average similarity is not always linear with the increase of δ as it can be seen in figure 5.7. In the case of HT, the average similarity stabilizes for values $\delta \in [0.4, 0.8]$ and in the case of HT+DM it grows oscillating.

Table 5.1: Results of average similarity and number of elements in the SDA diagram for each pathology for different values of δ

	HT		DM		HT+DM	
δ	Average sim.	# elements	Average sim.	# elements	Average sim.	# elements
0.3	0.76	35	0.68	23	0.76	36
0.4	0.88	39	0.79	35	0.76	36
0.5	0.86	43	0.81	42	0.87	54
0.6	0.84	44	0.88	61	0.93	70
0.7	0.86	47	0.88	62	0.93	83
0.8	0.84	49	0.90	67	0.87	86
0.9	0.95	55	0.95	75	0.91	90
1	1.00	103	1.00	159	1.00	229

Regarding the number of elements in the SDA diagram, we can observe that the three cases share the same trend. There is a, more or less, linear increase for values of δ lower than 1, and for $\delta = 1$ the number of elements grows drastically. This behavior is due to the fact that two action terms only have a similarity equal to 1 when they are exactly the same or, in the case of pharmacological action terms, if they belong to the same chemical group and have the same dose (see section 3.1.5.1). Therefore, it

is very difficult to generalize with this δ and the SDA diagram becomes very concrete (overfitting).

In general, we can conclude that $\delta = 0.3$ is usually the best choice because it provides diagrams with a low number of elements and the average similarity results are satisfactory.

5.4.2 Evolution of database adherence for hypertension during 2009

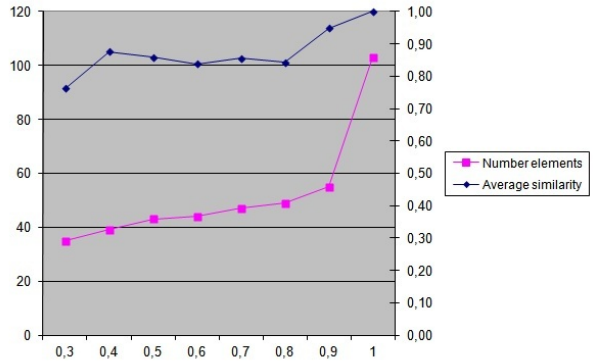
In order to conclude the tests about the adherence to the database, we have incrementally generated the diagram for HT with $\delta = 0.3$ during the year 2009. The purpose is to observe the evolution of the average similarity and the number of elements in the diagram for each month. Starting from January of 2009 we have incrementally updated an SDA diagram every single month with the new encounters that have taken place until December of the same year. The average similarity is calculated comparing the partial SDA diagram with the total database of patients of 2009 with the aim of observing how the SDA diagram progressively becomes complete.

The linear graph in figure 5.8 shows the results obtained.

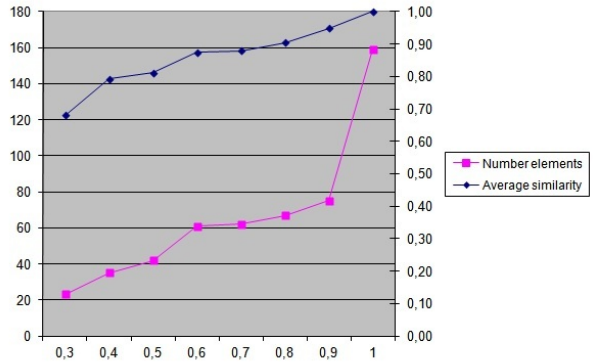
The average similarity lasts about eight months to stabilize. Observe that in August the average similarity is 0.72, which is almost equal to the result at the end of the year (0.76). Before these eight months there have not taken place enough encounters to represent all the variability of the treatment. Regarding the number of elements in the diagram, it grows faster and it is almost stabilized by May with 31 elements (only 4 less than the final diagram). These results suggest that in future updates of the diagram, the number of elements will keep more stable, while the average similarity will slowly increase.

5.5 Medical tests

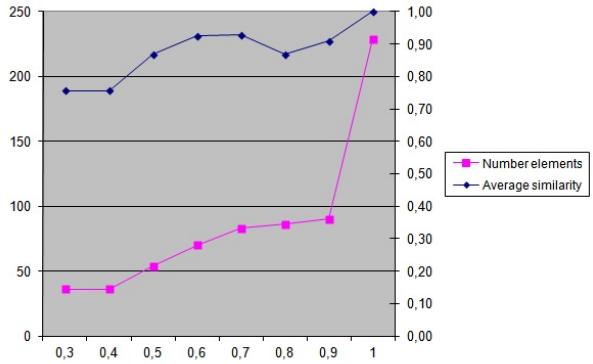
In this section we present the definitive SDA diagrams for HT, DM and the comorbidity of both diseases which have been generated from the hospital databases of SAGESSA during the year 2009 with the non-incremental approach. The background knowledge from the repository in section 3.3 has been used to guarantee medical correctness of the results. For each diagram we have chosen a $\delta = 0.3$ because, according to the health care professionals, it provided more readable diagrams without loss of medical quality.



(a) HT



(b) DM



(c) HT+DM

Figure 5.7: Linear graphs of average similarity and number of elements in the SDA diagram for each pathology for different values of δ

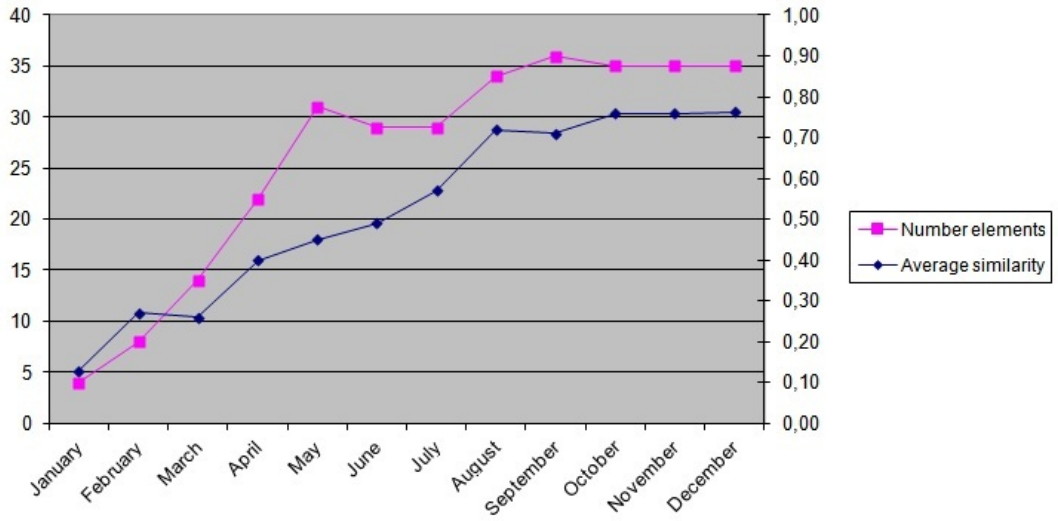


Figure 5.8: Linear graph of average similarity and number of elements in incremental SDA diagrams for hypertension during 2009

Here we analyze these diagrams from a medical point of view with the support of health care experts.

Moreover, we medically compare these diagrams with those generated with the knowledge-free approach in (BRLV12), reporting the advantages of our approach. We conclude the medical tests analyzing the evolution of the SDA diagram of HT generated in section 5.4.2.

5.5.1 SDA diagram and medical analysis for each pathology

Figure 5.9 depicts the SDA diagram for the treatment of HT that has been induced from a database of SAGESSA with all the patients treated in 2009.

The states of the diagram are based on the situation of the patient within the treatment of the disease. The level of control of the disease always appears in the first decisions after each state (except for patients taking 2 drugs).

According to the health care professionals, the SDA diagram reflects all the common situations in the treatment of HT. Due to the fact that the diagram is only based on pure hypertensive patients (without comorbidities), there are no situations of extreme treatment (e.g., patients taking 3 hypotensive drugs).

Figure 5.9: SDA diagram for the treatment of HT obtained from patients in 2009

We can observe some of the most typical cases represented in the SDA diagram. For example, when the patient is following none pharmacological treatment but his disease is controlled. In this case the SDA diagram indicates to continue with the same successful treatment. Another typical case reflected in the SDA diagram is when the patient is not following any kind of pharmacological treatment and his disease is not controlled so the SDA diagram indicates to start taking 1 hypotensive drug, and when the patient is taking 1 hypotensive drug and his disease is not controlled so he starts a treatment with 2 hypotensive drugs.

The SDA diagram does not include any information about drug replacement or dose modification, but this is because we have not used such low level of abstraction in the terminology of the actions.

Figure 5.10 depicts the SDA diagram for the treatment of DM that has been induced from a database of SAGESSA with the patients of 2009. The states of the diagram are based both on the situation of the patient within the treatment and the level of control of the disease.

When the diagram was validated by health care professionals, they reported that it lacked of several common clinical situations. A deeper analysis of the causes drove us to detect that in the data registered for the patients treated of DM in SAGESSA during 2009, there were no examples of all the common situations that can be found in the treatment of DM. Our algorithm must be seen as a tool to model health care treatments reflected in the input data, therefore it will not describe medical actions that are not present in the database.

The treatment of the DM without comorbidities has a low range of alternatives. Moreover, having grouped Oral Hypoglycemic Drugs (OHDs) in a unique action term, leads to a diagram which is small with a great proportion of states and very few decisions. The diagram is divided into three disconnected sub-diagrams. Health care professionals argued that there is a low ratio of patients taking insulins or OHDs that start taking both kind of drugs together. If these two situations were more common, the three sub-diagrams would be connected.

Another fact that has been detected is the absence of young people in the database, which causes OHDs to be the only pharmacological starting treatment.

For the combined treatment of HT and DM we have generated two different SDA diagrams with the patients of 2009, because we have two different versions of the

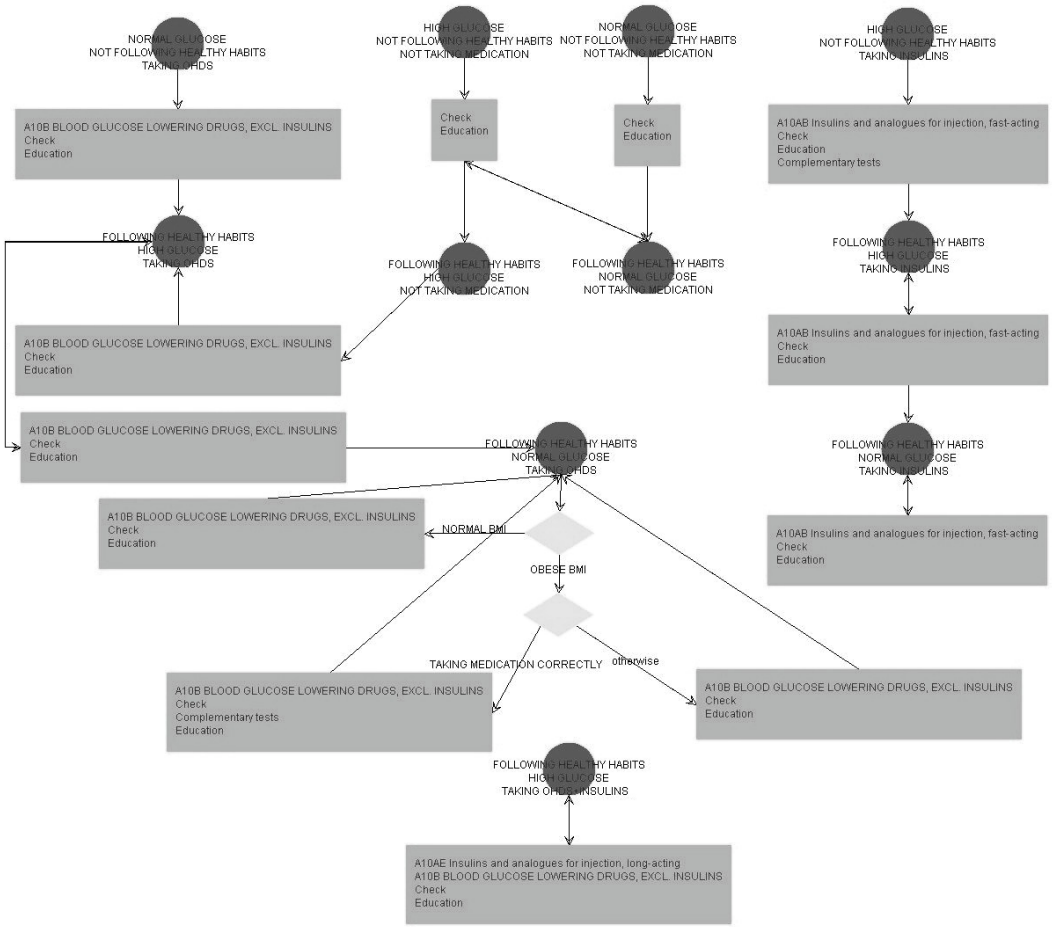


Figure 5.10: SDA diagram for the treatment of DM obtained from patients in 2009

background knowledge in the repository 3.3.3. The first one is depicted in figure 5.11.

In this first diagram, the states describe the situation of the patient within the treatment of the disease. The level of control of the diseases always appears in the first decisions after each state.

In general, this SDA diagram reflects all the common situations in the treatment of HT+DM.

Compared with the diagram of diabetes (see figure 5.10), the SDA diagram in figure 5.11 incorporates the transition between taking insulins and taking OHDs+insulins. However, there are several other transitions which our algorithm has not considered to include in the diagram, due to its low proportion in the database. Therefore, the diagram is composed by six disconnected sub-diagrams.

Our algorithm has been able to show evidence of several typologies of patients that do not appear in the database. For example, there are cases in which the treatment for HT is changed without knowing if the level of hypertension is controlled or not. This situation happens because there is a typology of patients, whose treatment for HT is not changed because it is not necessary, which is not registered in the database. A similar situation happens in some cases in which decisions are made without asking anything related to DM.

The other SDA diagram generated for the treatment of HT and DM is depicted in figure 5.12.

In this case, the states represent the level of control of the two diseases whereas the situation of the patient within the treatment always is left to appear in the first decisions after each state.

From the health care professionals' point of view, a logical tracking of the diagram can be made from any of the states. The SDA contains the four possible situations about the level of control of HT and DM and they are all connected.

This organization of the states leads to a SDA diagram for HT+DM which is preferred by health care professionals than the previous one.

5.5.2 Medical comparison with a knowledge-free approach

We have compared our diagrams with the ones obtained with the non-incremental approach that does not use background knowledge in (BRLV12). Due to the size of these

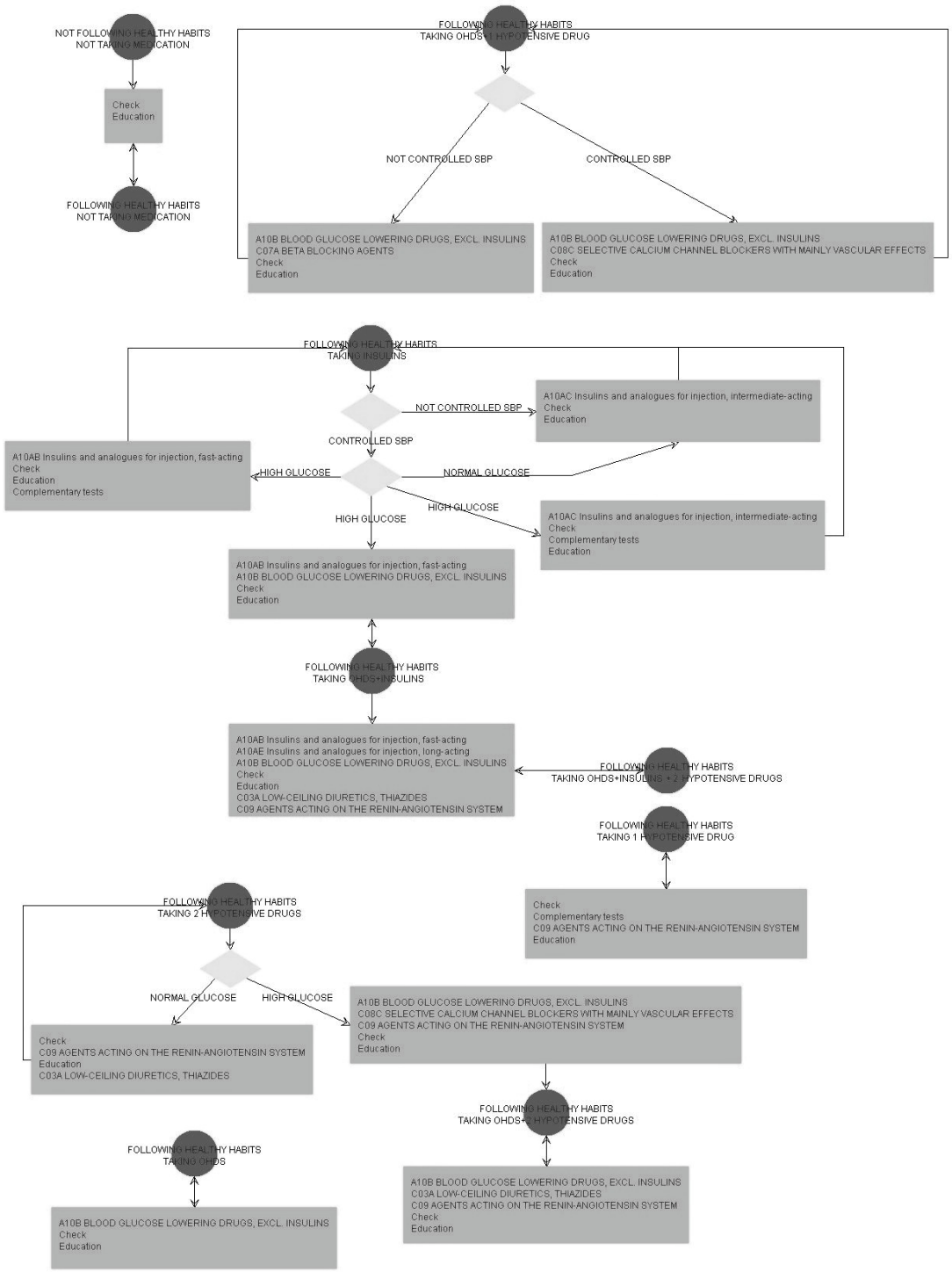


Figure 5.11: SDA diagram for the treatment of HT+DM obtained from patients in 2009 (states based on the stage of the treatment)

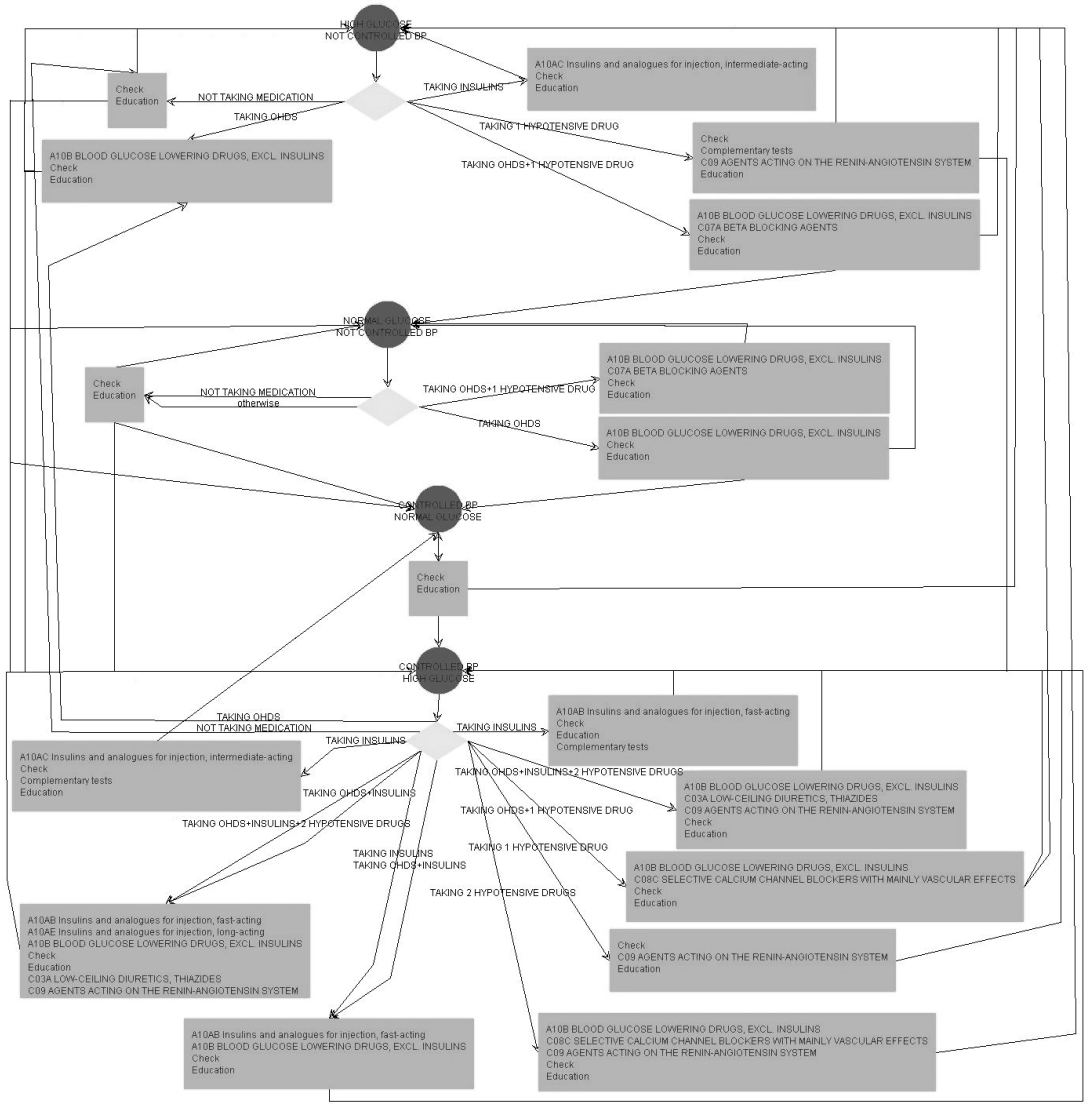


Figure 5.12: SDA diagram for the treatment of HT+DM obtained from patients in 2009 (states based on the level of control of the disease)

diagrams (more than 100 elements) we have considered not to include them in this document. The results obtained with this comparison have been included in (LVRC12a). The main conclusions reported in the paper are:

- The approach in (BRLV12) allows a syntactical adjustment of the number of states by means of a parameter, while our approach makes this adjustment in a semantical way, basically with the help of the background knowledge. For example, the number of states of the diagrams for HT obtained with the approach in (BRLV12) goes from 1 state (one state for all the encounters) to 57 states (one state for each different encounter) depending on the value given to the parameter. With our approach, the number of states for HT is always 4 because of the clinical priorities and constraints specified in the background knowledge.

- As we are not able to specify semantic decisions with the approach in (BRLV12), we cannot guarantee the absence of odd decisions in the SDA diagram. For example, one of the decisions in the diagram of HT of patients treated during 2009 asks whether the patient shows *ADEQUATE PHYSICAL ACTIVITY* or *NORMAL GLUCOSE*, which is a strange medical question to ask. This fact, together with the absence of priorities between decision terms, makes the resulting diagrams usually to be less comprehensible by physicians.

- The approach in (BRLV12) is not able to semantically compare actions, thus all the syntactically different actions are considered to be different even if they represent medically equivalent measures. This fact causes that the SDA diagram may contain unnecessary decisions and a great number of actions. For example, the diagram of the treatments of HT in 2009 has 67 syntactically different actions while our approach reduces them to 30 semantically different clinical actions.

As a conclusion, the approach that does not use background knowledge may be useful in controlled settings with a low number of state, decision and action terms as it shown in (BRLV12) where different SDA diagrams for HT are generated at a high level of abstraction for highly preprocessed data. In order to obtain medically correct diagrams in any setting, the use of background knowledge is essential. In addition, it also allows us to generate several diagrams with different intentionalities (using the

background knowledge related to the states) and different levels of abstraction for the same input data (modifying the value of δ).

5.5.3 Evolution of the SDA diagram for hypertension during 2009

Finally, we analyze the evolution of the SDA diagram of HT generated in section 5.4.2. Figure 5.13 depicts the diagram obtained using the encounters with patients during January of 2009.

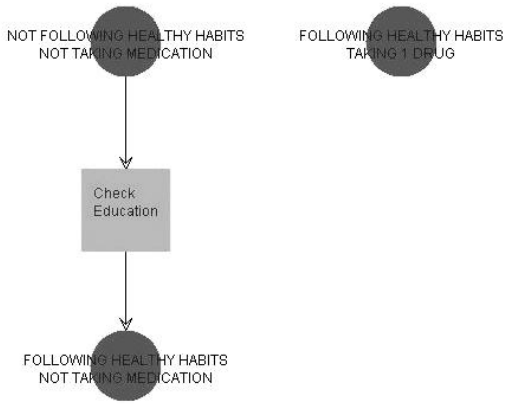


Figure 5.13: SDA diagram for the treatment of HT obtained from encounters in January of 2009

We can observe that only a few situations have taken place, leading to a very simple diagram. During this month, there were only situations of patients not following healthy habits and not taking medication which have been treated without the use of drugs. The state representing patients that are taking 1 drug appears in the diagram because some encounters with this kind of patient took place. However, no information is yet registered about the following encounters, therefore the algorithm is not able to indicate the treatment and evolution of these patients. After updating the previous diagram with the encounters of February we obtain the SDA diagram in figure 5.14.

With these new data, the diagram describes two non-deterministic clinical behaviors on patients not following healthy habits and not taking medication. The state representing patients that are taking 1 drug has been connected with the rest of the diagram showing a continuous evolution of the care. This connection comes represented by a clinical procedure described as a single clinical action that confirms the treatment

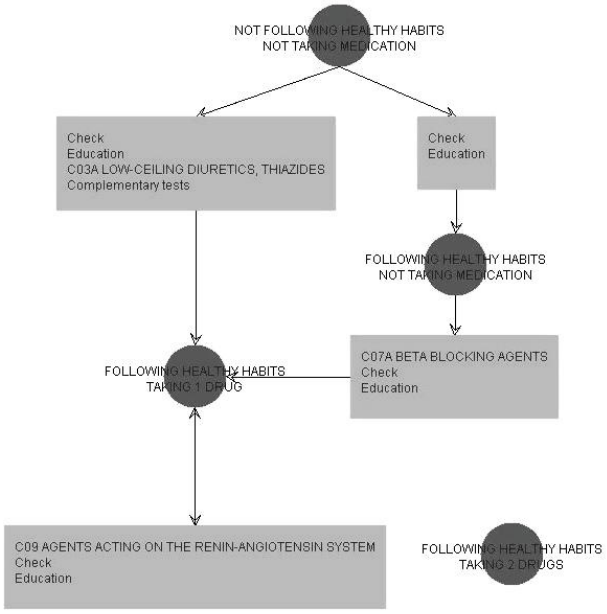


Figure 5.14: SDA diagram for the treatment of HT obtained from encounters in January-February of 2009

with a single drug. A new isolated state for patients taking 2 drugs has also appeared. Since these patients in the database are only visited once, the diagram is unable to reflect any evolution from this state to any other. The incremental update at the end of March leads to the diagram in figure 5.15.

This diagram contains the first decisions and increases treatment variability. The diagram is increased with new decisions and actions during the next months as we can see in figures 5.16, 5.17, 5.18 and 5.19.

By August, the diagram is almost equal to the final diagram (see figure 5.9). We have not included in the document the SDA diagrams of the next months because they are very similar. As it is expected, the final diagram obtained in December, after this incremental generation, is equal to the one generated with the non-incremental approach.

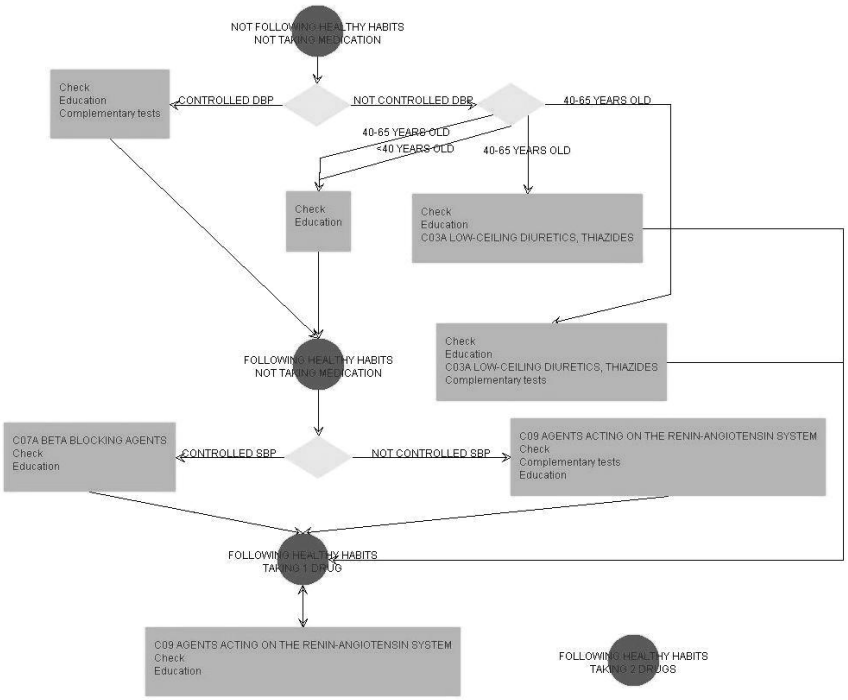


Figure 5.15: SDA diagram for the treatment of HT obtained from encounters in January-March of 2009

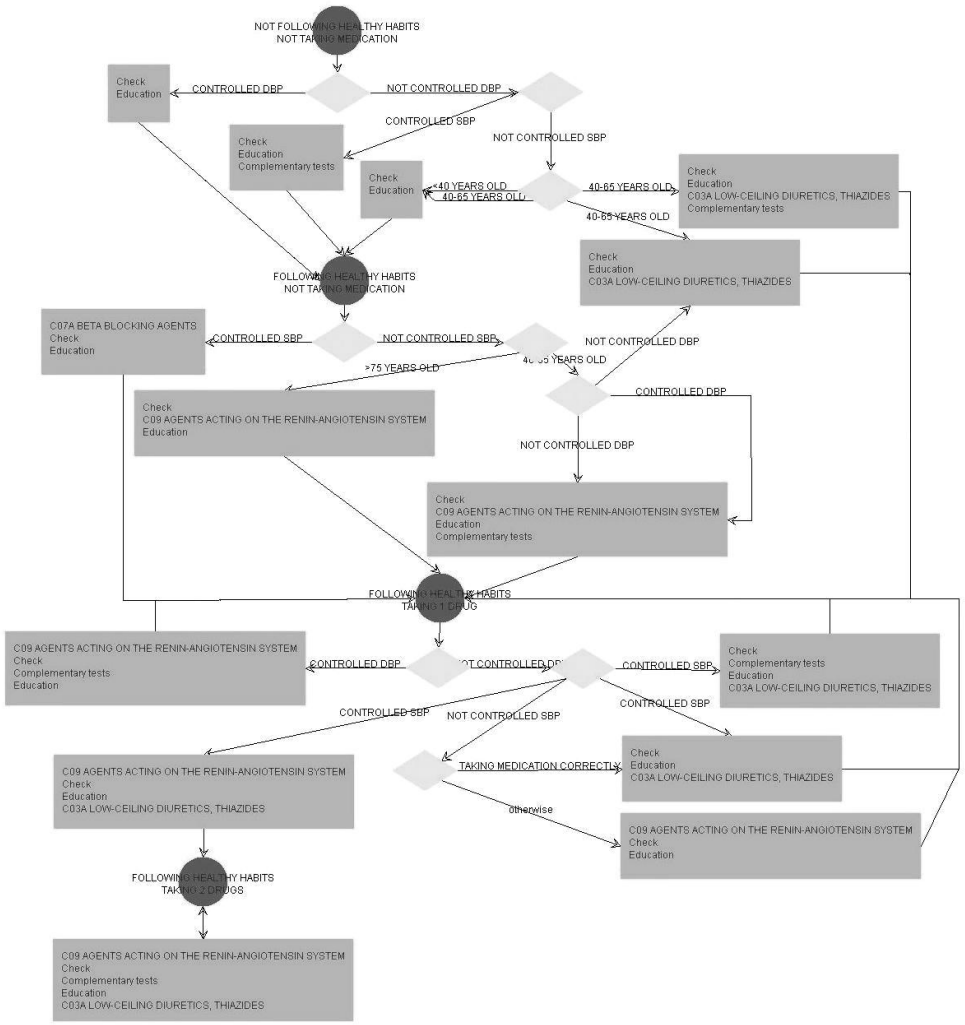


Figure 5.16: SDA diagram for the treatment of HT obtained from encounters in January-April of 2009

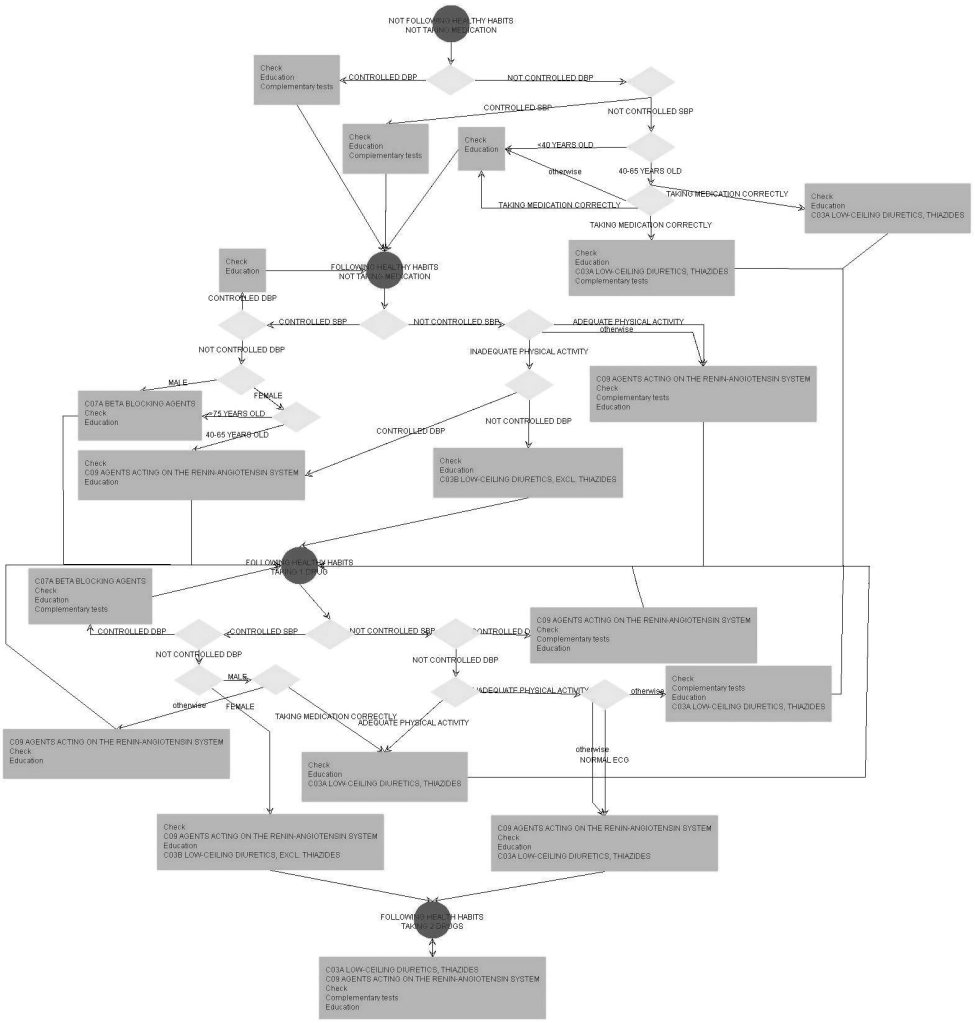


Figure 5.17: SDA diagram for the treatment of HT obtained from encounters in January-May of 2009

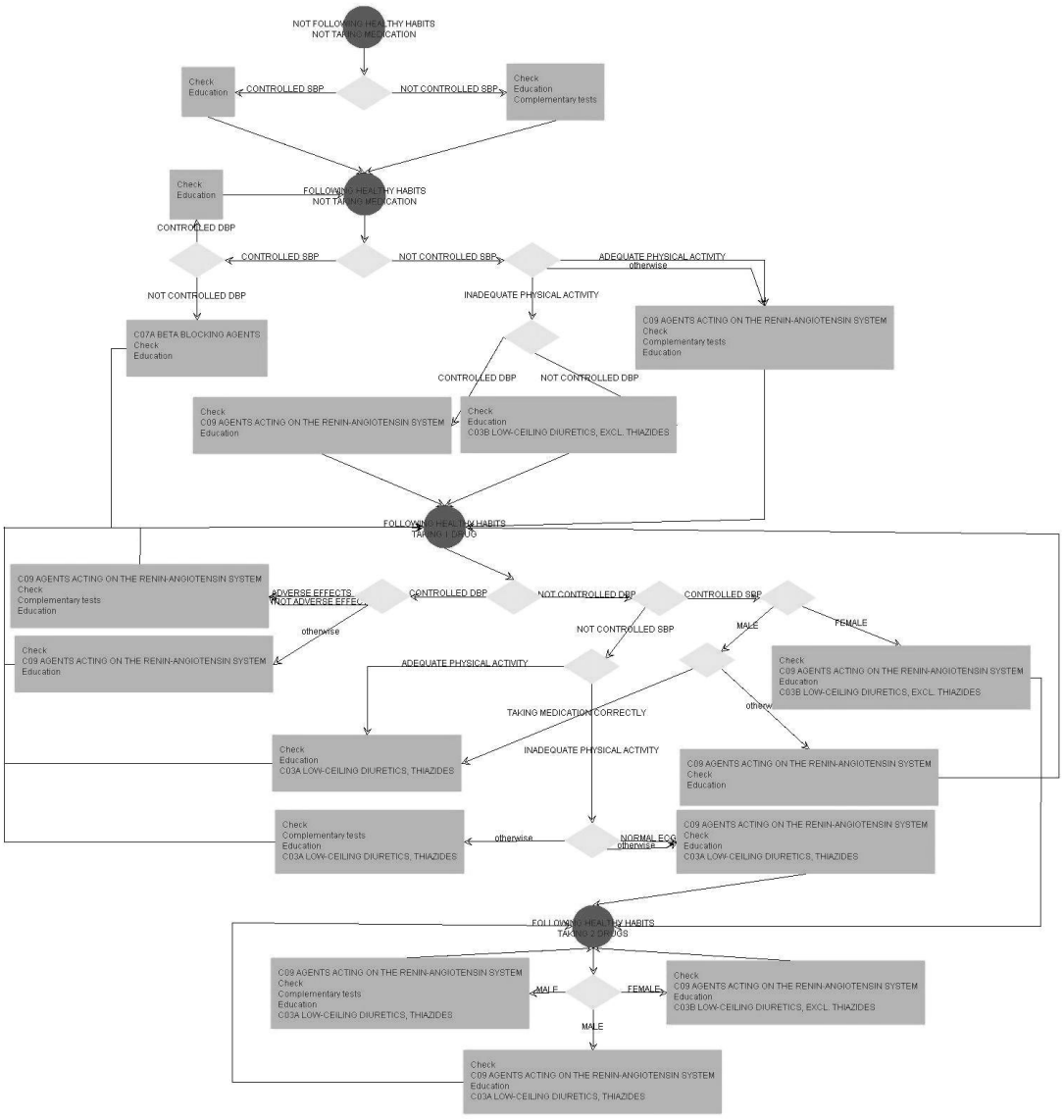


Figure 5.18: SDA diagram for the treatment of HT obtained from encounters in January-June of 2009

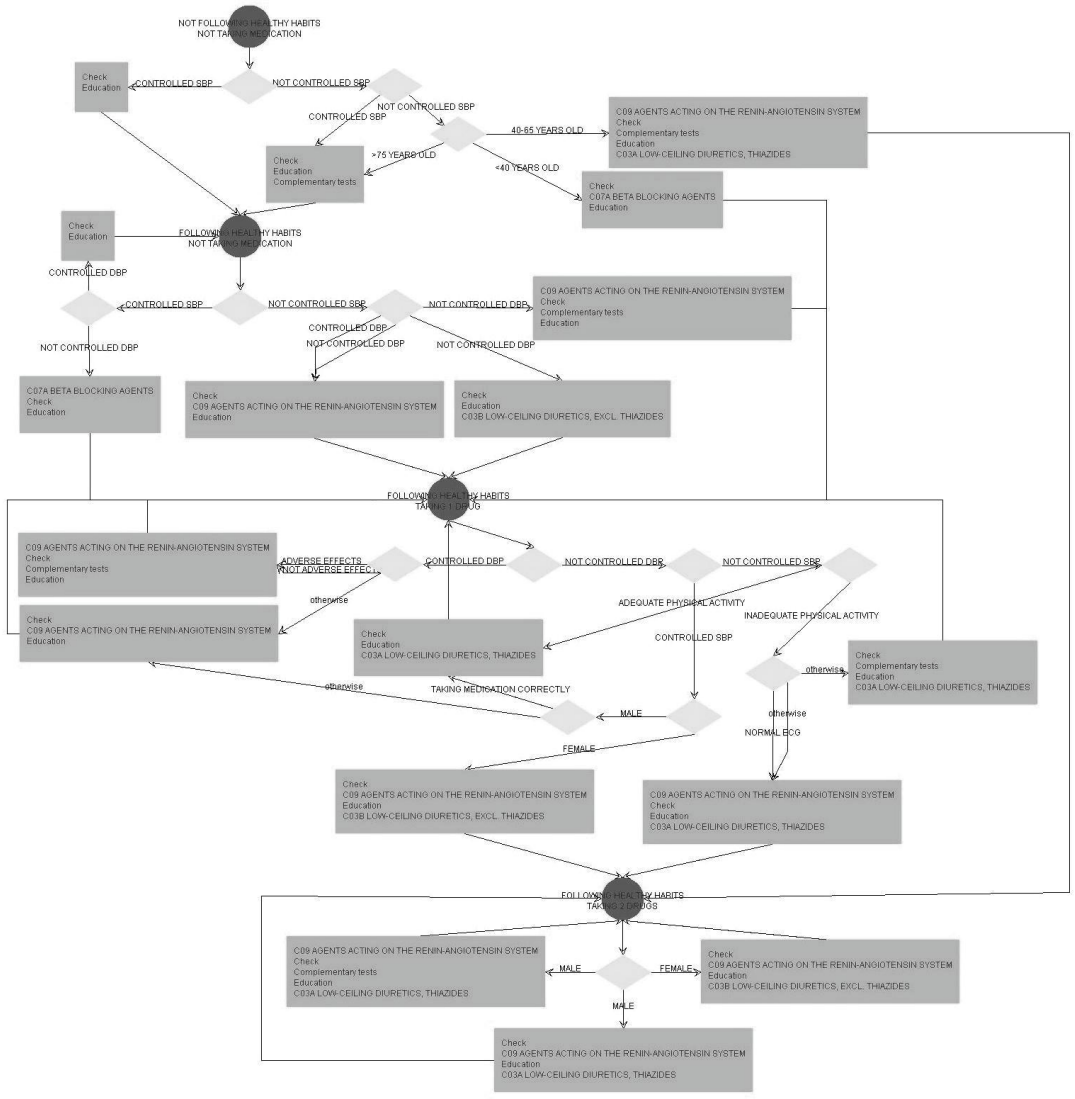


Figure 5.19: SDA diagram for the treatment of HT obtained from encounters in January-July of 2009

6

Conclusions

The main objective of this thesis has been to solve two of the main drawbacks of the current technologies to automatically induce CAs from hospital databases, which are:

- They are only based on statistical measures that do not necessarily respect medical criteria and semantics which can be essential to guarantee medical correct structures.
- They are not prepared to deal with the incremental arrival of new data which is worth to consider in health care, where hospital databases are constantly updated with new information about new arriving patients, and with the follow up of chronic patients.

Regarding the first drawback, we have studied several structures used to represent background knowledge and with the help of health care professionals we have proposed a formalization of all the background knowledge required to assist the automatic induction of medically correct and comprehensible CAs (contribution 1). This background knowledge includes constraints, preferences, semantic relationships and concept hierarchies, which condition the states, decisions and actions more appropriate to be part of the CAs.

Focusing our work in the diseases of HT, DM and the comorbidity of both diseases, we have constructed a repository of background knowledge. This knowledge is used to formalize the preferences and experience of health care professionals as well as the evidence-based knowledge contained in other resources like CPGs (SAG02; SAG03) or

the Anatomical Therapeutic Chemical (ATC) Classification System (fDSM) (contribution 2).

In order to solve the second drawback, we have proposed an incremental methodology to induce CAs which is partially inspired in some of the current incremental induction algorithms (Utg88; Utg89; UBC97). This new methodology includes two main procedures which are the identification of states and the determination of therapeutic sequences, which are integrated to generate CAs. In this work we represented CAs as SDA diagrams, according to the SDA knowledge model. This incremental methodology uses the formalized background knowledge to guarantee medically correct and comprehensible CAs (contribution 3).

The methodology has been evaluated with different kinds of tests. First of all, some tests have been done to verify the good performance at the level of background knowledge and incrementality. With respect to the former, the results show that the use of background knowledge does not imply an increase of temporal cost but a severe reduction of the time spent determining therapeutic sequences and, therefore, a reduction of the total duration of the process. The good performance of the incremental technology is confirmed after checking whether the methodology fulfills the three desirable goals of cost reduction, independence from the size and independence from the order.

A second kind of technological tests determined the adherence of the induced SDA diagrams to the hospital database that has been used to generated them. The tests have been done for HT, DM and HT+DM (comorbidity of both diseases) with different levels of abstraction in the terminology of the SDA diagrams. The results suggest that diagrams with a high level of abstraction are usually the best option because they have a low number of elements but keep satisfactory results on adherence to the database. A low level of abstraction slightly improves the adherence but it also increases the number of elements, and in some cases the SDA diagram can be unnecessarily concrete. The evaluation of the results by health care professionals reported that the diagrams were expressed at a good level of abstraction. More abstract diagrams would imply a lack of medical interest, while more detailed diagrams would cause the lack of alternative correct treatments not observed in the data and therefore a sense of uncompleteness in the diagrams. We have done another test following an incremental generation of a SDA diagram for HT during 2009, concluding that the number of elements in the diagram stabilizes faster than the adherence (four months and eight

months respectively) suggesting that, in future updates of the diagram, the number of elements will keep more stable, while the average similarity will slowly increase.

We have generated final SDA diagrams for the previous diseases to be analyzed from a medical point of view (contribution 4). In the case of HT the diagram obtained reflects all the common situations in the treatment of the disease. The health care professionals we have worked with were able to find the most typical situations in HT as for example, when the patient is not following any kind of pharmacological treatment and his disease is not controlled so treatment starts with taking 1 hypotensive drug. With regard to DM, with the generation of the diagram, we could determine that there were several situations which were not found in the database. Thus showing that our system could also be used to help physicians and health care managers to identify the casemix a health care center has to provide a service to and the set of treatments provided in that center. For the case of HT+DM we generated two diagrams, one with states based on the treatment of the diseases (i.e., the diagram showed the evolution of the treatment), and one with states based on the control of the diseases (i.e., the diagram showed the evolution of the diseases as the patient was treated). Thus showing not only that our methodology can be used to generate correct CAs that represent alternative views of the same treatments, but also that it is a valid methodology to represent the treatment of comorbidities. This last is particularly relevant since this sort of knowledge cannot be found explicitly represented since the CAs that appear in CPGs are restricted to one single disease. All these SDA diagrams have been compared with the knowledge-free approach in (BRLV12) and the results obtained show evidence that, in order to guarantee medically correct diagrams in any setting, the use of background knowledge is essential, and it is also very useful to generate several diagrams with different intentionalities and different levels of abstraction for the same input data. Finally, we have analyzed the evolution of the SDA diagram of HT during 2009 and confirmed that the diagram generated incrementally is equal to the one generated with the non-incremental approach.

The culmination of this thesis is another step forward in improving the automatic generation of clinical algorithms. The fact that the structures extracted from the hospital databases are medically correct and comprehensible will help the acceptance of this kind of methodologies by the medical community.

Moreover, being an incremental system, it will fit daily clinical practice allowing the representation of up-to-date procedural knowledge with acceptable costs. Therefore, it will be applicable in health care centers in order to supervise their medical procedures or to determine whether certain CPGs are being followed or not, or even to improve these guidelines with the knowledge extracted from the experience.

At the moment, the work on the development of this thesis has implied the publication of two papers in scientific journals of impact:

- In (BRLV12) we present a first approach to the automatic generation of SDA diagrams which does not consider background knowledge and does not work incrementally as a first step to the methodology presented in this thesis. In the paper we generate SDA diagrams for HT using the databases of SAGESSA and we study the deviations of the treatments with respect to official and predefined protocols and clinical algorithms, showing a high level of adherence to the treatment proposed by the National Heart Foundation of Australia and the Spanish Society for Hypertension with about 90.4% of coincident treatment.

- In (LVRB12) we propose a method to induce medically correct and comprehensible DTs based on some medical criteria included as background knowledge, using cost functions and partial orders. The results suggest that the method obtains DTs that physicians evaluate as more comprehensible and correct than the DTs obtained by previous approaches as they keep an equivalent accuracy.

The results of the thesis have been included in another journal paper (LVRC12a) which introduces all the formalizations related to the background knowledge, compares our methodology with the one without background knowledge and reports the conclusions that have been detailed in this document. The results about incrementality will give rise to another journal paper.

We have also presented two papers in international conferences:

- In (TRLV10) we present another approach to induce medically correct and comprehensible DTs which is not based on background knowledge structures. The paper proposes a slight variation of classical DTs, provides four quality ratios

to measure the medical correctness of a DT, and introduces an algorithm to induce DTs whose final decisions are both correct and the result of a sequence of observations with a medical sense.

- In (LVRC12b) we present the method to semantically compare clinical actions (see section 3.1.5). In the paper, the method is applied to analyze the data about the treatment of HT in a health care center in order to analyze feasible cost reductions after replacing medical interventions by their corresponding optimal, observed, dominant alternatives. This study shows that the use of this methodology reduces the average cost of each clinical encounter in €1.37.

Finally, we have published two research reports in the Universitat Rovira i Virgili:

- In (LVR11) we present cost functions and partial orders as a way to represent background knowledge and some mathematical operations to transform a cost function into a partial order (and vice versa) and also to combine several cost functions and partial orders into a common structure.
- In (LVR12) we present a hierarchy of medical criteria and we formalize their representation as cost functions and layered partial orders.

7

Future work

Medicine is an evolving science and, therefore, it is common that the protocols used to treat patients change over time, or even that new drugs or procedures are discovered causing variations in the physicians' daily practice. In the approach proposed in this thesis this behavior is partially considered. During the incremental learning of a SDA diagram we are able to change the background knowledge obtaining different results. For example, we can add new constraints between state terms or change their priorities so that the induced SDA diagram will divide the treatment into different new stages. We can even add new semantic decisions with a high priority that will lead to different therapeutic sequences. However, since our approach always considers the whole set of encounters with patients that have arrived over time, without giving higher priority to the newer ones, it is not able to forget the health care procedures that have become obsolete.

Incremental learning systems can be classified according to their memory model that dictates how to treat past training examples into three categories (MM00):

- *full instance memory*: the learner retains all past training examples.
- *partial instance memory*: the learner retains some of the past training examples.
- *no instance memory*: the learner retains none of the past training examples.

Our approach to induce SDA diagrams is an incremental learning system with full instance memory. Both the identification of states and the determination of therapeutic sequences store the whole set of encounters presented (though in an optimized way)

and it is always fully considered in these learning tasks. Other examples of incremental learning systems with full instance memory are GEM (RM88), ID5 (Utg88) and ITI (UBC97).

In order to deal with the situation described in the first paragraph, we are planning to modify our approach so that it forgets the encounters of patients that contain actions that have become obsolete. This forgetting behavior is also known as *aging* and it is usually related to partial instance memory (and also no instance memory) learning systems. There are at least three types of aging in the bibliography. The first type is the proximity-based aging like the one used by the system DARLING (Sal93). This partial instance memory algorithm initializes the weight of each new example to one and decays the weights of examples within a neighborhood of the new example. When an examples weight falls below a threshold, it is removed. A second type is the frequency-based aging which is used by the FAVORIT system (KK92a; KK92b). This system to induce decision trees with no instance memory, ages training examples either positively or negatively with respect to time (i.e., the newer the example is, the more important it becomes, and vice versa). If incoming training examples do not reinforce a nodes presence in the tree, then the nodes score decays. If the score falls below a threshold, then the algorithm forgets the node. Conversely, if incoming training examples continue to reinforce and revise the node, its score increases. If the score surpasses an upper threshold, then the nodes score is fixed and remains so. The last type of aging is time-based. The partial instance memory FLORA2 system (W96) is an example of this kind of aging. It selects a consecutive sequence of training examples from the input stream and forgets those examples that are older than a threshold, which is set adaptively. This system was designed to handle drifting concepts, so during periods when the system is performing well, the size of the window is increased and thus it keeps more examples. If the performance of the system drops, presumably due to some change in the target concepts, it reduces the size of the window and forgets the old examples to accommodate the new examples from the new target concept. As the systems concept descriptions begin to converge toward the target concepts, the size of the window increases again, as does the number of training examples maintained in partial memory.

A time-based aging similar to the one used by the FLORA2 system is the one that best suits our problem. In the future we are planning to incorporate it to our approach

to induce SDA diagrams in order to handle obsolete actions. Being A_1 the action used to treat a certain kind of patient, if new encounters of patients of this kind start using an alternative action A_2, the system may consider either that A_1 and A_2 are two alternative actions that coexist or that A_2 is a new action that will replace A_1. Therefore it will keep a window of encounters whose size will increase or decrease in order to determine if A_1 and A_2 are current treatments (type-1 non-determinism) or if A_1 has become obsolete.

We will also explore other applications of aging for the induction of SDA diagrams. For example, the removal of old encounters which essentially contain the same health care procedures that some newer ones, with the aim of reducing the spatial cost of the knowledge structures stored.

Apart from the incorporation of aging, we are also planning to keep experimenting with the parameters of the methodology in order to refine them, to construct alternative structures of background knowledge that allow the induction of SDA diagrams with different views of the same diseases studied in this thesis, and also to apply our methodology to other chronic diseases and comorbidities such as chronic heart failure, chronic obstructive pulmonary disease, ischaemic heart disease, and hyper-cholesterolemia.

Bibliography

[AP09] Arsham Alamian and Gilles Paradis. Clustering of chronic disease behavioral risk factors in canadian children and adolescents. *Preventive Medicine*, 48:493–99, 2009. 8

[BAL+01] R. Babuška, L. Alic, M.S. Lourens, A.F.M. Verbraak, and J. Bogaard. Estimation of respiratory parameters via fuzzy clustering. *Artificial Intelligence in Medicine*, 21:91–105, 2001. 8

[BBB+96] A. Burgun, G. Botti, O. Bodenreider, D. Delamarre, J.M. Levêque, B. Lukacs, D. Mayeux, M. Bremond, F. Kohle, M. Fieschi, and P. Le Beux. Methodology for using the umls as a background knowledge for the description of surgical procedures. *International Journal of Bio-Medical Computing*, 43(3):189–02, 1996. 25

[BBM02] Sugato Basu, Arindam Banerjee, and R. Mooney. Semi-supervised clustering by seeding. In *Proceedings of 19th International Conference on Machine Learning (ICML-2002)*, 2002. 28

[BBM+04a] Sugato Basu, A. Banjeree, E.R. Mooney, Arindam Banerjee, and Raymond J. Mooney. Active semi-supervision for pairwise constrained clustering. In *Proceedings of the 2004 SIAM International Conference on Data Mining (SDM-04)*, 2004. 28

[BBM04b] Sugato Basua, Mikhail Bilenko, and Raymond J. Mooney. Semi-supervised clustering for intelligent user management. In *Proceedings of the IBM Austin Center for Advanced Studies 5th Annual Austin CAS Conference, Austin, TX,*, February 2004. 28

[BBM04c] Mikhail Bilenko, Sugato Basu, and Raymond J. Mooney. Integrating constraints and metric learning in semi-supervised clustering. In *Proceedings of the 21 st International Conference on Machine Learning*, Banff, Canada, 2004. 28

[BCM+05] P.A. Bath, C. Craigs, R. Maheswaran, J. Raymond, and P. Willett. Use of graph theory to identify patterns of deprivation and high morbidity and mortality in public health data sets. *Journal of American Medical Informatics Association*, 12(6):630–41, 2005. 17

[BFOS84] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone. Classification and regression trees. Technical report, Wadsworth International Group, Belmont, CA, 1984. 33

[BGK+02] Jako S. Burgers, Richard Grol, Niek Klazinga, Marjukka Mäkelaä, and Joost Zaat. Towards evidence-based clinical practice: an international survey of 18 clinical guideline programs. *Int. Journal for Quality in Health Care*, 15(1):31–045, 2002. 1

[BJ06] Michael E Bales and Stephen B Johnson. Graph theoretic modeling of large-scale semantic networks. *Journal of Biomedical Informatics*, 39(4):451–464, Aug 2006. URL: `http://dx.doi.org/10.1016/j.jbi.2005.10.007`, `doi:10.1016/j.jbi.2005.10.007`. 14, 17

[BKG+05] A. Banerjee, C. Krumpelman, J. Ghosh, S. Basu, and R. J. Mooney. Model-based overlapping clustering. In *KDD '05: Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 532–37, New York, NY, USA, 2005. ACM Press. 7

[BM03] M. Bilenko and R.J. Mooney. Adaptive duplicate detection using learnable string similarity measures. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2003)*, pages 39–48, 2003. 28

[BP09] Olivier Bodenreider and Lee B. Peters. A graph-based approach to auditing rxnorm. *Journal of Biomedical Informatics*, 42(3):558 – 570, 2009. Auditing of Terminologies. URL: `http://www.sciencedirect.com/science/article/B6WHD-4W4TY17-1/2/b5b577e6e8d9b119af6602a5ea5b9fdc`, `doi:10.1016/j.jbi.2009.04.004`. 14, 17

[BRLV12] John A. Bohada, David Riaño, and Joan A. López-Vallverdú. Automatic generation of clinical algorithms within the state-decision-action model. *Expert Systems with Applications*, 39 (2012):10709–10721, 2012. `doi:http://dx.doi.org/10.1016/j.eswa.2012.02.196`. 2, 34, 40, 41, 126, 128, 130, 140, 144, 147, 157, 158

[BRR07] John A. Bohada, David Riaño, and Francis Real. Induction of partial orders to predict patient evolutions in medicine. In *AIME '07: Proceedings of the 11th conference on Artificial Intelligence in Medicine*, pages 489–499, Berlin, Heidelberg, 2007. Springer-Verlag. `doi:http://dx.doi.org/10.1007/978-3-540-73599-1_65`. 19, 20, 21, 22

[Bur98] C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):1–47, 1998. 9

[cC05] Gabriela Şerban and Alina Câmpan. Incremental clustering using a core-based approach. In *Computer and Information Sciences - ISCIS 2005*, volume 3733/2005 of *Lecture Notes in Computer Science*, pages 854–63. Springer Berlin / Heidelberg, 2005. 32

[CCFM97] Moses Charikar, Chandra Chekuri, Tomás Feder, and Rajeev Motwani. Incremental clustering and dynamic information retrieval. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 626–35, El Paso, Texas, United States, 1997. 31

[CCM03] D. Cohn, R. Caruana, and A. McCallum. Semi-supervised clustering with user feedback. Technical Report Tech. Report TR2003-1892, Cornell University, 2003. 28

[CH67] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–7, 1967. 9

[CH06] Yen-Liang Chen and Hui-Ling Hu. An overlapping cluster algorithm to provide non-exhaustive clustering. *European Journal of Operational Research*, 173(3):762–80, 2006. 7

[CL08] C. Chao and S. Liu. Diabetes mellitus treatment. *International Encyclopedia of Public Health*, page 153160, 2008. 2, 43

[Cle04] Martin L. Vrain C. Cleuziou, G. Poboc: an overlapping clustering algorithm. application to rule-based classification and textual data. In *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI-04)*, 2004. 7

[CN89] P. Clark and T. Niblett. The cn2 induction algorithm. *Machine Learning*, 3, 4:261–83, 1989. 6, 9

[COR05] Carlo Combi, Barbara Oliboni, and Rosalba Rossato. Merging multimedia presentations and semistructured temporal data: a graph-based model and its application to clinical information. *Artificial Intelligence in Medicine*, Volume 34 , Issue 2:89–12, 2005. 14, 15, 17

[Cra89] S.L. Crawford. Extensions to the cart algorithm. *International journal of man-machine studies*, 31:197–17, 1989. 33

[DH00] P. Domingos and G. Hulten. Mining high-speed data streams. In ACM Press, editor, *Proceedings KDD 2000*, pages 71–80, New York, NY, USA, 2000. 33

[DH04] S. Roberts D. Husmeier, R. Dybowski, editor. *Probabilistic Modelling in Bioinformatics and Medical Informatics.* Springer, 2004. 6

[Dun73] J. C. Dunn. A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters. *Journal of Cybernetics*, 3:32–57, 1973. 7

[dV96] Francisco Alte da Veiga. Structure discovery in medical databases: a conceptual clustering approach. *Artificial Intelligence in Medicine*, 8:473–91, 1996. 9, 32

[EKS+98] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Michael Wimmer, and Xiaowei Xu. Incremental clustering for mining in a data warehousing environment. In *Proceedings of the 24rd International Conference on Very Large Data Bases table of contents*, pages 323–33, 1998. 31

[Elk01] Charles Elkan. The foundations of cost-sensitive learning. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, 2001. 29

[FCPB07] Alberto Freitas, Altamiro Costa-Pereira, and Pavel Brazdil. Cost-sensitive decision trees applied to medical data. In *Data Warehousing and Knowledge Discovery*, volume 4654/2007 of *Lecture Notes in Computer Science*, pages 303–12. Springer Berlin / Heidelberg, 2007. 14, 26, 29

[fDSM] WHO Collaborating Centre for Drug Statistics Methodology. Anatomical therapeutic chemical classification system. http://www.whocc.no/atc. 23, 50, 61, 84, 156

[FGK+09] Michael R. Fellows, Jiong Guo, Christian Komusiewicz, Rolf Niedermeier, and Johannes Uhlmann. Graph-based data clustering with overlaps. In Lecture Notes in Computer Science, editor, *Computing and Combinatorics*, volume 5609/2009, pages 516–26. Springer Berlin / Heidelberg, 2009. 7

[Fis87] D. Fisher. Knowledge acquisition via incremental conceptual clustering. *Machine Learning*, 2:139–72, 1987. 8, 31

[FS99] Y. Freund and R. Schapire. Large margin classification using the perceptron algorithm. *Machine Learning*, 37:277–96, 1999. 9

[FYL+06] Zhuo Fang, Jiong Yang, Yixue Li, Qingming Luo, and Lei Liu. Knowledge guided analysis of microarray data. *Journal of Biomedical Informatics*, 39:401–11, 2006. 27

[GC00] C. G. Giraud-Carrier. A note on the utility of incremental learning. *AI Commun*, 13(4):215–24, 2000. 30

[GC03] F.A. Sonnenberg G.B. Chapman, editor. *Decision making in health care. Theory, psychology and applications*. Cambridge series on judgement and decision making. Cambridge University Press, 2003. 6, 9, 13

[GGR02] Russell Greiner, Adam J. Grove, and Dan Roth. Learning cost-sensitive active classifiers. *Artificial Intelligence archive*, 139(2):137–74, 2002. 29

[GKHAF09] Harsha Gurulingappa, Corinna Kolárik, Martin Hofmann-Apitius, and Juliane Fluck. Concept-based semi-automatic classification of drugs. *J. Chem. Inf. Model.*, 49:1986–992, 2009. 25

[GKK99] E. Han G. Karypis and V. Kumar. Chameleon: Hierarchical clustering using dynamic modeling. *IEEE Computer*, vol. 32, no. 8:6875, August 1999. 8

[GR93] J. Grimshaw and I. Russell. Achieving health gain through clinical guidelines. i: Developing scientifically valid guidelines. *QHC*, 2:243248, 1993. 1

[GRS00] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim. Rock: A robust clustering algorithm for categorical attributes. *Inf. Syst.*, vol. 25, no. 5:345–66, 2000. 8

[Had95] D.C. Hadorn. *Use of algorithms in clinical guideline development in Clinical Practice Guideline Development: Methodology Perspectives*, volume 95-0009. AHCPR, Rockville, MD, Jan 1995. 1, 34, 37

[HBBC08] Hatem Hamza, Yolande Belaïd, Abdel Belaïd, and Bidyut Baran Chaudhuri. Incremental classification of invoice documents. In *19th International Conference on Pattern Recognition - ICPR 2008*, 2008. 31

[Hua98] Z. Huang. Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data Mining and Knowledge Discovery*, 2:283–04, 1998. 7, 8

[IM08] D. Isern and A. Moreno. Computer-based execution of clinical guidelines: A review. *International Journal of Medical Informatics*, 77:787–808, 2008. 34

[Jam86] M. James. *Classification Algorithms*. Wiley-Interscience, 1986. 6

[JMF99] A.K. Jain, M.N. Murty, and P.J. Flynn. Data clustering: A review. *ACM Computing Surveys*, Vol. 31, No. 3, September 1999. 7

[JX06] Xiang Ji and Wei Xu. Document clustering with prior knowledge. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 405–12, 2006. 28

[KK92a] I. Krizakova and M. Kubat. Favorit: Concept formation with ageing of knowledge. *Pattern Recognition Letters*, 13:1925, 1992. 162

[KK92b] M. Kubat and I. Krizakova. Forgetting and aging of knowledge in concept formation. *Applied Artificial Intelligence*, 6:195206, 1992. 162

[KK08] Nimit Kumar and Krishna Kummamuru. Semisupervised clustering with metric learning using relative comparisons. *IEEE Transactions on Knowledge and Data Engineering*, 20(4):496–03, 2008. 28

[KKM02] Dan Klein, Sepandar D. Kamvar, and Christopher D. Manning. From instance-level constraints to space-level constraints: Making the most of prior knowledge in data clustering. In *Proceedings of the Nineteenth International Conference on Machine Learning*, pages 307–14, 2002. 28

[KMB+05] J. Steve Kammerer, Scott J.N. McNabb, Jose E. Becerra, Lisa Rosenblum, Nong Shang, Michael F. Iademarco, and Thomas R. Navin. Tuberculosis transmission in nontraditional settings: A decision-tree approach. *American Journal of Preventive Medicine*, 28(2):201–07, 2005. 13

[KN08] Mehmet Korürek and Ali Nizam. A new arrhythmia clustering technique based on ant colony optimization. *Journal of Biomedical Informatics*, 41:874–81, 2008. 8

[KP04] S. B. Kotsiantis and P. E. Pintelas. Recent advances in clustering: A brief survey. *WSEAS Transactions on Information Science and Applications*, 2004. 7

[KR90] L. Kaufman and P.J. Rousseeuw. Finding groups in data: An introduction to cluster analysis. *Wiley, New York*, 1990. 7, 8

[KZP06] S. Kotsiantis, I. Zaharakis, and P. Pintelas. Supervised machine learning: a review of classification techniques. *Artificial Intelligence Review*, 26(3):159–90, 2006. 9

[Lel94] Alain Lelu. Clusters and factors: neuronal algorithms for a novel representation of huge and highly multidimensional data sets. *New Approaches in Classification and Data Analysis. E. Diday, Y. Lechevallier & al. eds., Springer-Verlag, Berlin.*, pages 241–48, 1994. 7

[LGCM01] Timothy Langford, Christophe G. Giraud-Carrier, and John Magee. Detection of infectious outbreaks in hospitals through incremental clustering. In *Proceedings of the 8th Conference on AI in Medicine in Europe: Artificial Intelligence Medicine*, volume 2101 of *Lecture Notes In Computer Science*, pages 30–39, 2001. 32

[LGSD93] William J. Long, John L. Griffith, Harry P. Selker, and Ralph B. D'Agostino. A comparison of logistic regression to decision-tree induction in a medical domain. *Computers and Biomedical Research*, 26(1):74–97, 1993. 13

[LNM+09] Shengping Liu, Yuan Ni, Jing Mei, Hanyu Li, Guotong Xie, Gang Hu, Haifeng Liu, Xueqiao Hou, and Yue Pan. ismart: Ontology-based semantic query of cda documents. In *AMIA Annu Symp Proc.*, pages 375–79, 2009. 25

[LPM06] Bo Liu, Jiuhui Pan, and R I (Bob) McKay. Incremental clustering based on swarm intelligence. In *Simulated Evolution and Learning*, volume 4247/2006 of *Lecture Notes in Computer Science*, pages 189–96. Springer Berlin / Heidelberg, 2006. 31

[LTG+83] G. Landeweerd, T. Timmers, E. Gelsema, M. Bins, and M. Halic. Binary tree versus single level tree classification of while blood cells. *Pattern Recognition*, 16:571–77, 1983. 12

[Lu97] Yijun Lu. *Concept Hierarchy in Data Mining: Specification, Generation and Implementation*. PhD thesis, Simon Fraser University, Canada, December 1997. 22

[LV07] Joan A. López-Vallverdú. Sda lab, 2007. URL: `http://banzai-deim.urv.cat/repositories/repositories.html`. 126

[LvdGAH04] P. Lucas, L. van der Gaag, and A. Abu-Hanna. Bayesian networks in biomedicine and health-care. *Artificial Intelligence in Medicine*, 30(3):201–14, 2004. 6, 9

[LVR11] Joan Albert López-Vallverdú and David Riaño. Cost functions and partial orders as medical background knowledge: formalization and operations. research report deim-rr-11-003. Technical report, Universitat Rovira i Virgili, 2011. 87, 159

[LVR12] Joan Albert López-Vallverdú and David Riaño. Decision criteria in health-care and their representation with cost functions and layered partial orders. research report deim-rr-12-001. Technical report, Universitat Rovira i Virgili, 2012. 48, 100, 159

[LVRB12] Joan Albert López-Vallverdú, David Riaño, and John A. Bohada. Improving medical decision trees by combining relevant health-care criteria. *Expert Systems with Applications*, 39(14):11782–11791, 2012. `doi:http://dx.doi.org/10.1016/j.eswa.2012.04.073`. 2, 14, 21, 22, 26, 29, 30, 43, 99, 100, 101, 103, 158

[LVRC07] Joan Albert López-Vallverdú, David Riaño, and Antoni Collado. Increasing acceptability of decision trees with domain attributes partial orders. In *CBMS '07: Proceedings of the Twentieth IEEE International Symposium on Computer-Based Medical Systems*, pages 569–574, Washington, DC, USA, 2007. IEEE Computer Society. `doi:http://dx.doi.org/10.1109/CBMS.2007.59`. 2, 14, 21, 22, 29, 43

BIBLIOGRAPHY

[LVRC12a] Joan Albert López-Vallverdú, David Riaño, and Antoni Collado. Background knowledge to improve the induction of clinical algorithms. *Expert Systems with Applications (To be submitted)*, 2012. 60, 147, 158

[LVRC12b] Joan Albert López-Vallverdú, David Riaño, and Antoni Collado. Detecting dominant alternative interventions to reduce treatment costs. In LNAI, editor, *Knowledge Representation for Health-Care 2011*, volume 6924, pages 131–144, 2012. 2, 14, 25, 58, 106, 159

[LWY04] Jinze Liu, Wei Wang, and Jiong Yang. A framework for ontology-driven subspace clustering. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining table of contents*, pages 623–28, Seattle, WA, USA, 2004. 14, 25

[LXH+10] Shih-Fang Lin, Ke-Ting Xiao, Yu-Ting Huang, Chung-Cheng Chiu, and Von-Wun Soo. Analysis of adverse drug reactions using drug and drug target interactions and graph-based methods. *Artificial Intelligence in Medicine*, Volume 48 , Issue 2-3:161–66, 2010. 17

[LYWZ04] Charles X. Ling, Qiang Yang, Jianning Wang, and Shichao Zhang. Decision trees with minimal costs. In *Proceedings of the twenty-first international conference on Machine learning*, page 69, Banff, Alberta, Canada, 2004. 14, 26, 29

[MAEB04] R.J.Q. McNally, F.E. Alexander, O.B. Eden, and J.M. Birch. Little or no spacetime clustering found amongst cases of childhood lymphoma in north west england. *European Journal of Cancer*, 40:585–89, 2004. 8

[MCKD+06] Robert Moskovitch, Shiva Cohen-Kashi, Uzi Dror, Iftah Levy, Amit Maimon, and Yuval Shahar. Multiple hierarchical classification of free-text clinical guidelines. *Artificial Intelligence in Medicine*, 37(3):177–90, 2006. 14, 25

[McQ67] J. McQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–97, 1967. 7, 8

[MM00] Marcus A. Maloof and Ryszard S. Michalski. Selecting examples for partial memory learning. *Machine Learning*, 41:2752, 2000. 161

[MPM10] Snehasis Mukhopadhyay, Mathew Palakal, and Kalyan Maddu. Multi-way association extraction and visualization from biological text documents using hyper-graphs: Applications to genetic association studies for diseases. *Artificial Intelligence in Medicine*, 49(3):145–54, July 2010. 16, 17

[MS99] Francesco Masulli and Andrea Schenone. A fuzzy clustering based segmentation system as support to diagnosis in medical imaging. *Artificial Intelligence in Medicine*, 16(2):129–47, 1999. 8

[MSL+08] R. Mans, H. Schonenberg, G. Leonardi, S. Panzarasa, A. Cavallini, S. Quaglini, and W.M.P. van der Aalst. Process mining techniques: an application to stroke care. In S.K. Andersen SK et al., editor, *eHealth Beyond the Horizon*, pages 573–578. IOS Press, 2008. 1, 40

[MSRS07] Maryann N. Mugo, Craig S. Stump, Priya G. Rao, and James R. Sowers. *Hypertension*, chapter Chapter 34 - Hypertension and Diabetes Mellitus, pages 406–417. 2007. 2, 43

[MSSvdA08] R.S. Mans, M.H. Schonenberg, M. Song, and W.M.P. van der Aalst. Process mining in healthcare: A case study. In *Proceedings of Health-inf 2008, International Conference on Health Informatics, INSTICC,*, volume 1, pages 118–125, 2008. 1, 40

[MvdAP07] N. Mulyar, W.M.P. van der Aalst, and M. Peleg. A pattern-based analysis of clinical computer-interpretable guideline modeling languages. *Journal of American Medical Informatics Association*, 14(6):781–87, 2007. 34

[NML06] Shu-Kay Ng, Geoffrey J. McLachlan, and Andy H. Lee. An incremental em-based learning approach for on-line prediction of hospital resource utilization. *Artificial Intelligence in Medicine*, 36(3):257–67, 2006. 33

[Nor89] Steven W. Norton. Generating better decision trees. In *Proceedings of the 11th international joint conference on Artificial intelligence*, volume 1, pages 800–05, Detroit, Michigan, 1989. 29

[Nuñ91] Marlon Nuñez. The use of background knowledge in decision tree induction. *Machine Learning*, 6(3):231–50, 1991. 29

[ohhs03] U.S. Department of health and human services. The seventh report of the joint national committee on prevention, detection, evaluation, and treatment of high blood pressure, 2003. URL: `http://banzai-deim.urv.net/repositories/Bibliography/HTeng.pdf`. 2, 43

[PAKS09] Mor Peleg, Nuaman Asbeh, Tsvi Kuflik, and Mitchell Schertz. Ontoclusta methodology for combining clustering analysis and ontological methods for identifying groups of comorbidities for developmental disorders. *Journal of Biomedical Informatics*, 42:165–75, 2009. 9

[PKSR02] V. Podgorelec, P. Kokol, B. Stiglic, and I. Rozman. Decision trees: an overview and their use in medicine. *J Med Syst*, 26(5):445–63., 2002. 6, 9, 13

[Poo07] Wouter Poortinga. The prevalence and clustering of four major lifestyle risk factors in an english adult population. *Preventive Medicine*, 44:124–28, 2007. 8

[PPT+02] M. Phil, M. Peleg, S. Tu, A.A. Boxwala, R.A. Greenes, and V.L. Patel et al. Representation primitives, process models and patient data in computer-interpretable clinical practice guidelines. *International Journal of Medical Informatics*, 68:59–70, 2002. 34

[PS03] S. Pemmaraju and S. Skiena. *Computational Discrete Mathematics*. 2003. 18

[PTB+03] M. Peleg, S. Tu, J. Bury, P. Ciccarese, J. Fox, and R.A. Greenes et al. Comparing computer-interpretable guideline models: A case-study approach. *Journal of American Medical Informatics Association*, 10(1):52–68, 2003. 34

[Qui86] J.R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986. 11, 12, 32

[Qui93] J.R. Quinlan. *C4.5: Programs for Machine Learning*. San Mateo, CA., USA, 1993. 11, 12

[RD00] Oldemar Rodríguez and Edwin Diday. Pyramidal clustering algorithms in iso-3d project. In *Proceedings of the 4th European Conference on Principles and Practice of Knowledge Discovery in Databases, PKDD'00*, 2000. 7, 8

[RGO+04] José C. Ribeiro, Sandra Guerra, José Oliveira, Antonio Teixeira-Pinto, Jos W.R. Twisk, José A. Duarte, and Jorge Mota. Physical activity and biological risk factors clustering in pediatric population. *Preventive Medicine*, 39:596–01, 2004. 8

[RH01] J. Robert and L.C.J. Howlett. Radial basis function networks. *New Advances in Design*, 2, 2001. 9

[Ria07] David Riaño. The sda model: A set theory approach. In *Proc. Twentieth IEEE Int. Symp. Computer-Based Medical Systems CBMS '07*, pages 563–568, 2007. doi:10.1109/CBMS.2007.110. 2, 34, 40

[Ria10] David Riaño. The eoc data model (v1.0). Technical report, Universitat Rovira i Virgili, http://banzai-deim.urv.net/repositories/Documents/EOC.doc, November 2010. 2, 33

[RLVT08] David Riaño, Joan Albert López-Vallverdú, and Samson Tu. Mining hospital data to learn sda* clinical algorithms. In *Knowledge Management for Health Care Procedures*, volume 4924 of *Lecture Notes in Computer Science*, pages 46–61. Springer, 2008. 2, 33, 40, 41

[RM88] R. Reinke and R. Michalski. *Machine Intelligence 11*, chapter Incremental learning of concept descriptions: A method and experimental results. Oxford: Clarendon Press, 1988. 162

BIBLIOGRAPHY

[RRLV+12] David Riaño, Francis Real, Joan Albert López-Vallverdú, Fabio Campana, Sara Ercolani, Patrizia Meccoci, Roberta Annicchiarico, and Carlo Caltagirone. An ontology-based decision support system for the care of chronically ill patients. *Journal of Biomedical Informatics*, 45(3):429–446, 2012. 27

[RWL+08] Xiaogang Ruan, Jinlian Wang, Hui Li, Rhoda E. Perozzi, and Edmund F. Perozzi. The use of logic relationships to model colon cancer gene expression networks with mrna microarray data. *Journal of Biomedical Informatics*, Volume 41, Issue 4:530–43, 2008. 17

[SA04] Irena Spasić and Sophia Ananiadou. Using automatically learnt verb selectional preferences for classification of biomedical terms. *J Biomed Inform*, 37(6):483–497, Dec 2004. URL: `http://dx.doi.org/10.1016/j.jbi.2004.08.002`, `doi:10.1016/j.jbi.2004.08.002`. 21

[SAG] Hospital consortium sagessa. URL: `http://www.grupsagessa.com/`. 2, 61

[SAG02] Grup SAGESSA. *Guia de maneig de la diabetis.* 2002. URL: `http://www.grupsagessa.com/documents/menupai/pai_diabetis-catala.pdf`. 51, 61, 84, 155

[SAG03] Grup SAGESSA. *Guia de maneig de la hipertensió.* 2003. URL: `http://www.grupsagessa.com/documents/menupai/pai_hipertensio-catala.pdf`. 51, 61, 84, 155

[Sal93] M. Salganicoff. Density-adaptive learning and forgetting. In *Proceedings of the Tenth International Conference on Machine Learning*, pages 276–83, San Francisco, CA, 1993. Morgan Kaufmann. 162

[SAM05] Diego Sona, Paolo Avesani, and Robert Moskovitch. Helping physicians to organize guidelines within conceptual hierarchies. In *Artificial Intelligence in Medicine*, volume 3581/2005 of *Lecture Notes in Computer Science*, pages 141–45. Springer Berlin / Heidelberg, 2005. 14, 25

[SC00] Ramón Sangüesa and Ulises Cortés. Prior knowledge for learning networks in non-probabilistic settings. *International Journal of Approximate Reasoning*, 24:103–20, 2000. 13

[Sch06] G. Schwartz. Health care guideline: Hypertension diagnosis and treatment, Oct 2006. Last accessed June 22, 2010. URL: `http://www.icsi.org/guidelines_and_more/gl_os_prot/cardiovascular/hypertension_4/hypertension_diagnosis_and_treatment__11.html`. 38

[SF86] J.C. Schlimmer and D. Fisher. A case study of incremental concept induction. In Morgan Kaufmann, editor, *Proceedings of the Fifth National Conference on Artificial Intelligence*, pages 496–01, Philadelphia, PA, 1986. 32

[SfMDM92] Committee on Standardization of Clinical Algorithms Society for Medical Decision Making, editor. *Medical Decision Making*, volume 12, chapter Proposal for Clinical Algorithm Standards, pages 149–54. 1992. 1, 34, 37, 38, 40

[SGS98] R. Rastogi S. Guha and K. Shim. Cure: An efficient clustering algorithm for large databases. In *Proc. ACM SIGMOD Int. Conf. Management of Data*, page 7384, 1998. 8

[SH01] Gabriel L. Somlo and Adele E. Howe. Incremental clustering for profile maintenance in information gathering web agents. In *Proceedings of the fifth international conference on Autonomous agents table of contents*, pages 262–69, Montreal, Quebec, Canada, 2001. 31

[Shi97] R.N. Shiffman. Representation of clinical practice guidelines in conventional and augmented decision tables. *Jour*, 4:382–93., 1997. 6

[SKP03] Miyoung Shin, Eun Mi Kang, and Seon Hee Park. Automatically finding good clusters with seed k-means. *Genome Informatics*, 14:326–27, 2003. 28

[SL91] S. Rasoul Safavian and David Landgrebe. A survey of decision tree classifier methodology. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(3):660–74, 1991. 11

[SMC+00] Cynthia J. Sims, Leslie Meyn, Rich Caruana, R. Bharat Rao, Tom Mitchell, and Marijane Krohn. Predicting cesarean delivery with decision tree models. *American Journal of Obstetrics and Gynecology*, 183(5):1198–206, 2000. 13

[SRG86] J.C. Schlimmer and Jr. R.H. Granger. Incremental learning from noisy data. *Machine Learning*, 1:317–54, 1986. 33

[SSK04] Dan Simovici, Namita Singla, and Michael Kuperberg. Metric incremental clustering of nominal data. In *Proceedings of the Fourth IEEE International Conference on Data Mining*, pages 523–26, 2004. 31

[SvLTO02] A. Jantine Schuit, A. Jeanne M. van Loon, Marja Tijhuis, and Marga C. Ocké. Clustering of lifestyle risk factors in a general adult population. *Preventive Medicine*, 35:219–24, 2002. 8

[SW48] C. Shannon and W. Weaver. *The mathematical theory of communication*. Urbana, IL, USA, 1948. 11

[SWW94] Von-Wun Soo, Jan-Sing Wang, and Shih-Pu Wang. Learning and discovery from a clinical database: An incremental concept formation approach. *Artificial Intelligence in Medicine*, 6(3):249–61, 1994. 25

[SY06] Charles X. Lingand Victor S. Sheng and Qiang Yang. Test strategies for cost-sensitive decision trees. *IEEE Transactions on Knowledge and Data Engineering archive*, 18(8):1055–067, 2006. 14, 26, 29

[Tan93] Ming Tan. Cost-sensitive learning of classification knowledge and its applications in robotics. *Machine Learning*, 13(1):7–33, 1993. 29

[TB99] Luis Talavera and Javier Béjar. *Advances in Intelligent Data Analysis*, volume Volume 1642/1999 of *Lecture Notes in Computer Science*, chapter Integrating Declarative Knowledge in Hierarchical Clustering Tasks, pages 211–22. Springer Berlin / Heidelberg, 1999. 28

[TBK09] Luis Tari, Chitta Baral, and Seungchan Kim. Fuzzy c-means clustering with prior biological knowledge. *Journal of Biomedical Informatics*, 42:74–81, 2009. 8, 27

[Tin98] Kai Ming Ting. Inducing cost-sensitive trees via instance weighting. In Lecture Notes In Computer Science, editor, *Proceedings of the Second European Symposium on Principles of Data Mining and Knowledge Discovery*, volume 1510, pages 139–47, 1998. 29

[TRLV10] Pere Torres, David Riao, and Joan Albert López-Vallverdú. Inducing decision trees from medical decision processes. In LNAI, editor, *Knowledge Representation for Health-Care*, volume 6512, pages 40–55, 2010. 158

[Tur00] P.D. Turney. Types of cost in inductive concept learning. In *Workshop on Cost-Sensitive Learning at the Seventeenth International Conference on Machine Learning (WCSL at ICML-2000)*, Stanford University, California, 2000. 28, 29

[Tur09] K.J. Turner. Abstraction and analysis of clinical guidance trees. *Journal of Biomedical Informatics*, 42:237–50, 2009. 6, 9

[TZL96] R. Ramakrishnan T. Zhang and M. Livny. Birch: An efficient data clustering method for very large databases. In *Proc. ACM SIGMOD Conf. Management of Data*, pages 103–14, 1996. 8

[UBC97] P.E. Utgoff, N.C. Berkman, and J.A. Clouse. Decision tree induction based on efficient tree restructuring. *Machine Learning*, 29:5–44, 1997. 32, 100, 156, 162

[UFM05] Antti Ukkonen, Mikael Fortelius, and Heikki Mannila. Finding partial orders from unordered 0-1 data. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 285–93, Chicago, Illinois, USA, 2005. ACM New York, NY, USA. 21

[Utg88] P. Utgoff. Id5: An incremental id3. In Morgan Kaufmann Publishers, editor, *Fifth International Conference on Machine Learning*, pages 107–20, 1988. 32, 100, 156, 162

[Utg89] P.E. Utgoff. Incremental induction of decision trees. *Machine Learning*, 4:161–86, 1989. 32, 100, 156

[Utg94] P. Utgoff. An improved algorithm for incremental induction. Technical Report UM-CS-1994-007, University of Massachusetts, Amherst, MA, USA, 1994. 30

[VBL09] Aida Valls, Montserrat Batet, and Eva M. López. Using experts rules as background knowledge in the clusdm methodology. *European Journal of Operational Research*, 195:864–75, 2009. 28

[vdAvDH+03] W.M.P. van der Aalst, B.F. van Dongen, J. Herbst, L. Maruster, G. Schimm, and A.J.M.M. Weijters. Workflow mining: A survey of issues and approaches. *Data & Knowledge Engineering*, 47:237–267, 2003. 1, 38, 40

[VdCFL07] M. Velikova, N. de Carvalho Ferreira, and P. Lucas. Bayesian network decomposition for modeling breast cancer detection. In Springer, editor, *Artificial Intelligence in Medicine (AIME 2007)*, volume 4594 of *LNAI*, pages 346–50, Amsterdam, The Netherlands., 2007. 6, 9

[VMFC09] Marco Vassura, Luciano Margara, Piero Fariselli, and Rita Casadio. A graph theoretic approach to protein structure selection. *Artificial Intelligence in Medicine*, Volume 45 , Issue 2-3:229–37, 2009. 14, 17

[VSS+09] J.M. Valderas, B. Starfield, B. Sibbald, C. Salisbury, and M. Roland. Defining comorbidity: Implications for understanding health and health services. *Annals of Family Medicine*, 7(4):357–63, 2009. 1

[W96] G. Widmer and M. Kubat (1996). Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23:69101, 1996. 162

[WC10] Kiri Wagstaff and Claire Cardie. Clustering with instance-level constraints. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 1103–110, 2010. 27

[WCRS01] Kiri Wagstaff, Claire Cardie, Seth Rogers, and Stefan Schroedl. Constrained k-means clustering with background knowledge. In *Proceedings of the Eighteenth International Conference on Machine Learning*, pages 577–84, 2001. 27

[Wil05] Scott B. Wilson. A neural network method for automatic and incremental learning applied to patient-dependent seizure detection. *Clinical Neurophysiology*, 116(8):1785–795, 2005. 33

[WS06] Kieran J. White and Richard F. E. Sutcliffe. Applying incremental tree induction to retrieval from manuals and medical texts. *Journal of the American Society for Information Science and Technology archive*, 57(5):588–00, 2006. 33

[XNJR03] E.P. Xing, A.Y. Ng, M.I. Jordan, and S. Russell. Distance metric learning, with application to clustering with side information. *Advances in Neural Information Processing Systems*, 15:505–12, 2003. 28

[XW05] Rui Xu and Donald Wunsch. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, Vol. 16, No. 3, May 2005. 7

[XW12] Rong Xu and QuanQiu Wang. A knowledge-driven conditional approach to extract pharmacogenomics specific druggene relationships from free text. *Journal of Biomedical Informatics*, 2012. `doi:http://dx.doi.org/10.1016/j.jbi.2012.04.011`. 14, 17

[ZD98] Blaž Zupan and Sašo Džeroski. Acquiring background knowledge for machine learning using function decomposition: a case study in rheumatology. *Artificial Intelligence in Medicine*, 14:101–17, 1998. 27

[Zha00] G. Zhang. Neural networks for classification: a survey. *IEEE Transactions on Systems, Man, and Cybernetics*, Part C 30(4):451–62, 2000. 9

Declaration

I herewith declare that I have produced this paper without the prohibited assistance of third parties and without making use of aids other than those specified; notions taken over directly or indirectly from other sources have been identified as such. This paper has not previously been presented in identical or similar form to any other examination board.

The thesis work was conducted from 2009 to 2012 under the supervision of Dr. David Riaño at Universitat Rovira i Virgili.

Tarragona,
15 October 2012