An Autonomous Algorithm for Generating and Merging Clinical Algorithms

Francis Real and **David Riaño**[1]

Abstract. Procedural knowledge in medicine uses to come expressed as isolated sentences in Clinical Practice Guidelines (CPG) that describe how to act in front of specific health-care situations. Although CPGs gather all evidence available on concrete medical problems, their direct application has been proved to have some limitations. One of these limitations occurs when they have to be applied on co-morbid patients suffering from several simultaneous and mutually related diseases. In such cases, health-care professionals have to follow the indications of multiple CPGs and solve their interactions as they appear in the treatment of concrete patients. Clinical Algorithms (CA) are schematic models of the procedures appearing in a CPG. They are used to organize and summarize the recommendations contained in CPGs. Here, we extend a knowledge-based algorithm to merge CAs with a machine learning procedure to relax the knowledge dependence of that algorithm. The resulting algorithm has been tested on health-care date provided by the SAGESSA Group on hypertension patients. The results obtained prove that it is a good approach to the generation of CA from data though several improvements at the levels of prediction and medical interpretation are possible.

1 INTRODUCTION

In health-care, procedural knowledge exists in multiple tasks as in the process of diagnosing a disease or in the application of a long-term therapy. All the evidence generated on this sort of knowledge for a particular disease is reported on a Clinical Practice Guideline (CPG). CPGs are defined as systematically developed statements to assist practitioners and patient decisions about appropriate health care for specific circumstances [2]. Surprisingly in this definition, the facts that CPGs are systematically developed and that CPGs are for specific circumstances may contradict their final purpose, which is to assist professional decisions. So, on the one hand systematic development in medicine is evidence-based, therefore, when there is not evidence on how to act under certain circumstances, the CPG has a knowledge gap that the physician has to fill with personal experience or consulting other colleagues whenever a patient under such circumstances arrives.

On the other hand CPGs use to be specific for one disease, which is called primary disease, and it may contain indications on how to act if the patient has other diseases, which are considered secondary in the CPG. However, nowadays, the most frequent patient is one with several important diseases that require simultaneous attention. That is to say, patients that require the simultaneous application of several CPGs. Whereas covering knowledge gaps in a CPG is a purely medical task, the simultaneous application of several guidelines affecting a patient is a task that can be systematized with computer intelligence techniques that merge CPGs [9, 5]. All these techniques are sustained on the representation of the procedural knowledge of the CPGs with formal languages that computers are able to manage (e.g., Protege [8], or SDA [7]).

Our approach to merging CPGs is sustained on the SDA language that represents the health-care procedural knowledge in the CPGs as clinical algorithms (CA) [4]. CAs can be found as part of CPGs [1], can be the result of a knowledge engineering process [3], or can be derived from health-care data [6]. The merging of CAs is based on the idea of finding out the basic knowledge units in each one of the CAs we want to merge, and then combine them into a single CA that gathers all the procedural knowledge scattered across the initial CAs. One of the main drawbacks of this approach is the need of complex and extensive expert knowledge to support one of the steps of the process. For each patient condition and action performed, this knowledge has to indicate what is the short-term expected evolution of that patient condition after applying that action. When there are several diseases involved, patient conditions and their evolution may be difficult to both evaluate and foresee by medical doctors. In such cases medical knowledge becomes complex because it cannot be found in CPGs, and large because the the number of alternative conditions of co-morbid patients may grow geometrically.

Here, we propose a machine learning procedure to carry out the merging of CAs that does not need the above mentioned complex and extensive expert knowledge. This procedure is sustained on the experience accumulated in health-care information systems about the treatment of comorbid patients.

In section 2 the merge of CAs is briefly introduced. Section 3 contains the description of the methodology to obtain CAs using medical data. Section 4 introduces two experiments with real data to generate different CAs. Finally, some conclusions are provided in section 5.

2 MERGING CAs

Clinical algorithms are schematic models of the clinical decision pathway described in a CPG. These algorithms combine health-care actions with decision points in a sequential process that represents the long-term treatment of a particular disease [4]. The SDA model [7], SDA standing for State-Decision-Action, extends the above idea of CA with the concept of patient state to describe alternative conditions of the patient that require differential treatments, and also as a way of determining the feasible evolution of the patients across the states as they are treated of the disease. States are used as possible starting points in the application of the CA.

A methodology for merging CAs in the SDA model was proposed

[1] Research Group on Artificial Intelligence, Rovira i Virgili University, Spain, email: {francis.real, david.riano}@urv.net

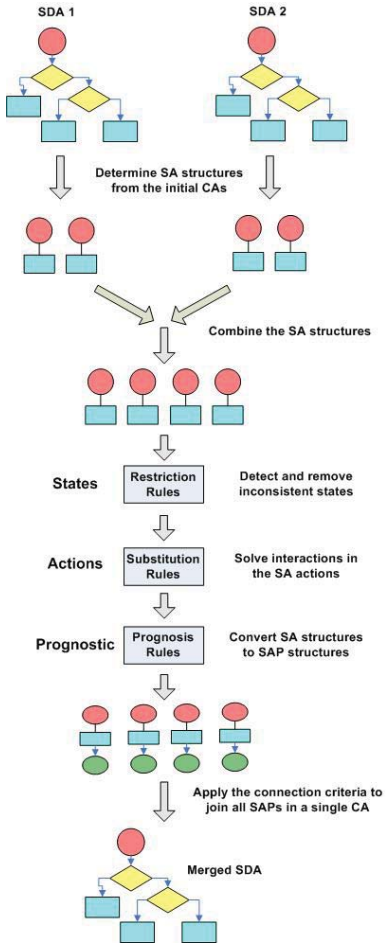

Figure 1. SDA merging methodology

in [5]. Figure 1 summarizes this methodology as a sequence of transformations that go from the initial CAs to the merged CA.

This merging procedure is based on a limited number of domain variables. These variables are the basic components of the intermediate state-action (SA) and state-action-prognosis (SAP) structures that are basic to the merging procedure which consists of the following steps:

1. Determine SA structures from the initial CAs
2. Combine the SA structures
3. Detect and remove inconsistent states
4. Solve interactions in the SA actions
5. Convert SA structures to SAP structures
6. Apply the connection criteria to join all SAPs in a single CA

The next subsections discuss these issues in more detail.

2.1 Domain variables

The health-care terms in the CAs are expressed as state and action variables. State variables comprise all the important aspects that may be used to describe the condition of the patients, whereas the action variables are those elements that describe the activities carried out during the treatment of the patient (e.g. counsels, prescriptions, test requests, etc.). For example, the treatment of hypertension can be described with the state variables *DBP*[2], *SBP*[2], *cholesterol* (all of them with values *normal*, *high*, and *very high*), *bad lifestyle* and *obesity* (as Boolean variables); and with the action variables *drug prescription*, and *change lifestyle*. Detailed reference to the sort of drugs is also possible extending the action variables.

The merging process described here is based on the idea that the domain variables of the merged CA are taken from the domain variables of the respective CAs merged. From a health-care point of view, it could be the case that the resulting CA would require the use of new variables. This case has not been considered in this work.

2.2 SA and SAP structures

The merging procedure consists in the decomposition of complex CAs in the basic SA structures. Each SA structure represents a knowledge of the sort *if-then* where the if part is the SA state, and the then part is the SA action. For example, the SA with S=*bad lifestyle*, *DBP high*, and A=*change lifestyle* proposes a change in the patient's lifestyle when a bad lifestyle and a high DBP is observed.

In the short-term evolution of a patient a SA can be extended to a SAP structure by introducing the expected evolution after an action A is performed on a patient with state S. In this paper, the concept short-term refers to the time between two consecutive patient-professional encounters. For example, the SAP with S=*bad lifestyle*, *DBP high*, and A=*change lifestyle*, and P=*DBP normal* indicates that the DBP of a patient with bad lifestyle will move from high to normal if the patient changes his lifestyle. On the contrary, long-term evolution refers to the medical changes happened between non consecutive encounters.

2.3 Combination of SA states

Two or more states can be combined to form a new state that contains all the state variables of the initial states. This sort of combination must not satisfy any of the constrains represented as a predefined set of restriction rules. A restriction rule is a subset of variables that cannot be observed simultaneously in the state of a patient. For example, if obesity and annorexy are state variables it is not be possible to have a patient state with both of them true.

Any state that satisfies a restriction rule is considered inconsistent, and the SA that contains it is removed from the set of SAs.

2.4 Combination of SA actions

Some action variables as drug prescriptions may have interactions. These interactions and the way they are solved are expressed by means of substitution rules. A substitution rule is a tuple (S, A_1, A_2) where S represents a patient condition as a set of state variables, and A_1 and A_2 are respectively the actions before and after the interactions are solved in the medical context described by S. Both A_1 and A_2 are represented as sets of action variables. For example, the substitution rule $(\{diabetes,\ hypertension\}, \{beta\text{-}blocker\}, \{ACEi^3\})$ indicates that hypertensive diabetic patients that are treated with beta-blockers must change their medication to ACEi.

Every time two or more actions have to be combined in the merging procedure, all the substitution rules are used to detect and solve the feasible interactions.

[2] DBP stands for Diastolic Blood Pressure
SBP stands for Systolic Blood Pressure
[3] ACEi stands for angiotensin-converting enzyme inhibitors.

2.5 Foreseeing short-term prognosis

SA structures represent instant decisions where patients fulfilling S are treated as A indicates. However, CAs are essentially sequential and they reflect long-term decisions in time. In order to be able to convert SA structures in a CA we construct some intermediate structures called SAPs that capture the concept of short-term decision. A SAP (S, A, P) enlarges an SA (S, A) with the introduction of a predictive component that indicates what is the expected sort-term evolution of a patient in state S which is applied the treatment A.

A prognosis rule is defined as a tuple (S_p, A_p, S'_p) where S_p and S'_p are subsets of state variables, and A_p is a subset of action variables. This kind of rules represent the expert knowledge that is used to calculate the P components of SAP structures with the following procedure:

```
Procedure Transform SA to SAP (S, A, PR) is
   P := S;
   repeat
      for each (Sp, Ap, S'p) in PR do
         if (Sp ⊂ S) and (Ap ⊂ A)
         then P := (P \ Sp) ∪ S'p
      end;
   until P does not change;
   return (S, A, P);
end.
```

A prognosis rule (S_p, A_p, S'_p) is applicable to a SAP structure (S, A, P) if S and A include S_p and A_p, respectively. When the rule is applied, a new SAP structure (S, A, P') is obtained with $P' = (P \setminus S_p) \cup S'_p$. That is to say, all the variables in P that are in S_p are replaced by the variables in S'_p.

For each SA structure (S, A), the above procedure starts generating a SAP structure (S, A, S), and the prognosis rules (S_p, A_p, S'_p) are repeatedly applied, as previously indicated, until the SAP structure does not change. The order of application of prognosis structures is the one they are provided by the health-care expert.

2.6 Connecting SAPs

The last of the merging steps is the combination of all the obtained SAP structures in a single CA. The new CA is the result of applying the connecting steps described in [6] and it represents a treatment which is the combination of several treatments.

3 USING MEDICAL DATA TO OBTAIN CAs

In the merging procedure explained in section 2, one of the main difficulties is to obtain medical knowledge from experts in the form of restriction, substitution and prognosis rules. An alternative to the introduction of this sort of knowledge by human experts is to use artificial intelligence algorithms to induce this knowledge from medical data. In this section we explore the induction of prognosis rules, and leave the induction of restriction and substitution rules for future work.

The induced rules will be used to transform SA structures into SAP structures as it is explained in section 2.5. In this new approach, unlike in the merging procedure introduced in section 2, the SA structures are not determined from the initial CAs, but automatically detected by the inductive algorithm in the data. The procedure to obtain the final CA is then composed of the following steps:

1. Determine the state and action variables from the medical data
2. Construct the data matrix
3. Find out the SA structures
4. Obtain the SAP structures
5. Apply the connection criteria to join all SAPs in a single CA

All these steps are described in the next subsections.

3.1 Medical data

Medical data uses to come expressed in terms of specific vocabulary that is useful to represent concrete health-care activities, but which may lose their appropriateness when they have to be used in the description of CAs, where concepts have to be more general. This specific vocabulary is used to fill the data structures that contain the information about the treatment of patients. Data structures in this work are represented in figure 2 where each patient contains a list of encounters, each one with information about observations and healthcare activities in the encounter.

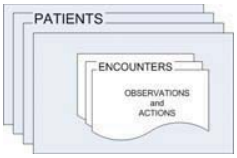

Figure 2. Data model used in tests

Before the medical data is ready for the construction of CAs, the specific vocabulary in the data has to be converted to the state and action variables we want our final CA to contain. This process is made automatic with the introduction of transformation operators that allows the data to be filtered or generalized. These operators are of either the form (*state variable, observation, condition, value*) or (*action variable, activity*). In the first case all the observations that fulfill the condition in the data is converted to the state variable with the given value. In the second case, all the activities in the data are converted to the action variable indicated. For example, (DBP, TAD, > 100, VERY-HIGH) is an operator that converts all the TAD[4] values greater than 100 to the state Boolean variable DBP-VERY-HIGH.

After the application of the transformation operators, all the state and action variables define a data matrix whose columns are the state variables twice and the action variables once. The first state variables define the current state of the patient, the second is the short-term evolution of the patient (i.e., prognosis variables) after the activities indicated by the action variables are applied. Each row of the matrix represents a different encounter.

3.2 Obtaining SA structures from the data matrix

An *action block* is each one of the different combination of values of the action variables in the data matrix. Therefore, several encounters may share the same action block. We express $(S_i = true)_A$ as the number of encounters whose state variable S_i is true in the action block A, $(S_i = true)$ as the number of encounters whose state variable S_i is true, and N_A as the number of encounters in the action block A. Then, equation 1 is used to calculate the relevance of any state variable S_i in the definition of any action block A.

[4] TAD stands for the Spanish word for DBP

This relevance is based on the idea that the most frequent a state variable appears in an action block and the less it appears in other action blocks, the more relevant it is to define the patients that are treated with that action block.

$$\alpha_i = \sqrt{\frac{(S_i = true)_A^2}{(S_i = true)N_A}} \quad (1)$$

Let S_κ^A be the set of the κ state variables with greater relevance for an action block A. Then, (S_κ^A, A) describes a feasible SA structure representing N_A^κ encounters with patients in state S_κ^A and with treatment A. N^κ is the number of encounters with patients in state S_κ^A and any treatment. For all of such feasible SA structures we define β_A^κ in equation 2 as the selection criterion for the best SA structure to be generated. The rest of structures are not generated.

$$\beta_A^\kappa = N_A^\kappa \left(\frac{N_A^\kappa}{N^\kappa}\right)^2 \kappa \quad (2)$$

If (S, A) is the generated SA structure, we remove all the encounters with patients in state S from the data matrix and repeat the process until the number of encounters in the matrix is reduced to a predefined percentage.

3.3 Obtaining prognosis from data

Each one of the SA structures obtained in the previous section has to be transformed into an SAP structure with the help of the data matrix. As described in section 3.1, the data matrix contains information about state, action, and prognosis variables. Given a SA structure (S, A), the expression $(P_i = true)_{(S,A)}$ is the number of encounters with patients in state S which receive treatment A and with prognosis variable P_i=true, and $(P_i = true)$ the number of encounters with prognosis variable P_i=true. These numbers are combined in equation 3 to calculate the prognosis capability of P_i in (S, A).

$$\gamma_i = (P_i = true)_{(S,A)} \frac{(P_i = true)_{(S,A)}}{(P_i = true)} \quad (3)$$

The prognosis P in the SAP structure contains the first κ prognosis variables with higher influence value γ, κ being a predefined parameter of the system.

3.4 Apply connecting conditions

Once the SAP structures are induced, they are joined in a CA with the connection criteria that is explained in [6].

4 TESTS

To evaluate this work we propose an experiment with real data. Our objective is to obtain two different CAs from a dataset on the treatment of hypertension. The first CA is made to differentiate between when a patient requires drug treatment and when the patient requires changes in his lifestyle. On the contrary, the second CA is designed to show how to apply the different pharmacological treatments.

4.1 Data description

The medical data used in this experiment has been provided by SAGESSA group[5]. It consists of a set of 28 patients with hypertension, equivalent to a total of 684 encounters with a minimum of 2 encounters and a maximum of 47 encounters per patient.

[5] http://www.grupsagessa.com/

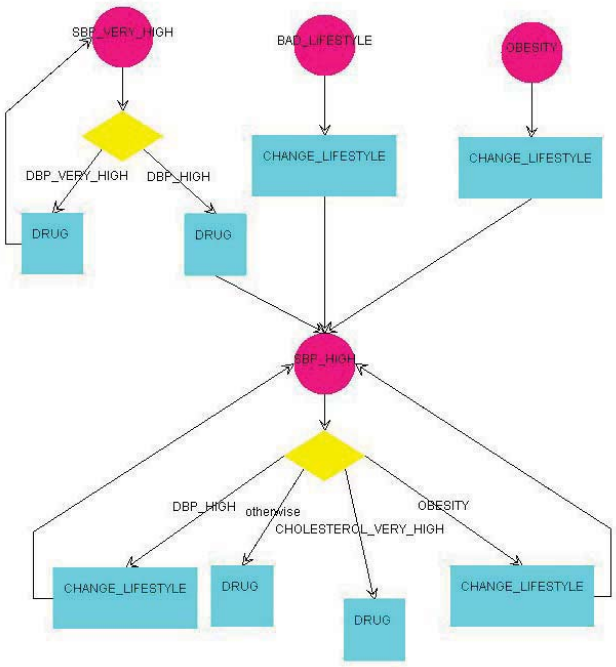

Figure 3. CA of the general treatment of hypertension

In the source data there are 100 different state variables, but only 43 appear more that 10 times, and 169 action variables, of which 134 correspond to drug prescriptions, 18 to request of new tests, and 17 to changes in the patient's lifestyle.

4.2 Results

For the first CA we use the state variables *DBP*, *SBP*, *cholesterol*, *bad lifestyle* and *obesity*; and the action variables *drug prescription*, and *change lifestyle*. The medical data is filtered and generalized (see section 3.1) to obtain a data matrix with 131 out of 684 encounters, all of them containing at least one state variable, one action variable and one prognosis variable.

When the methodology described in section 3 is applied we obtain eight SA structures and eight prognosis rules. Figure 3 shows the resulting CA.

The second CA takes all the state variables provided in the medical data, and the action variables *ACEi*, *Alpha-blockers*, *Beta-blockers*, *Diuretics*, *CCB*[6] and *ARB*[6]. After the filtering and generalization process the data matrix is reduced to 38 encounters.

4.3 Evaluation

In order to evaluate the results we use an statistical criterion which is based on the classification of the encounters in the CAs. Table 1 shows the results of this evaluation where the number of encounters that can be included in at least one of the states of the CA appears as *classified*, and among them the number of encounters that also show a treatment equivalent to the sort of treatment indicated in the CA as *good classified*.

[6] CCB stands for Calcium-Channel Blockers
ARB stands for Angiotensin-II Receptor Antagonists

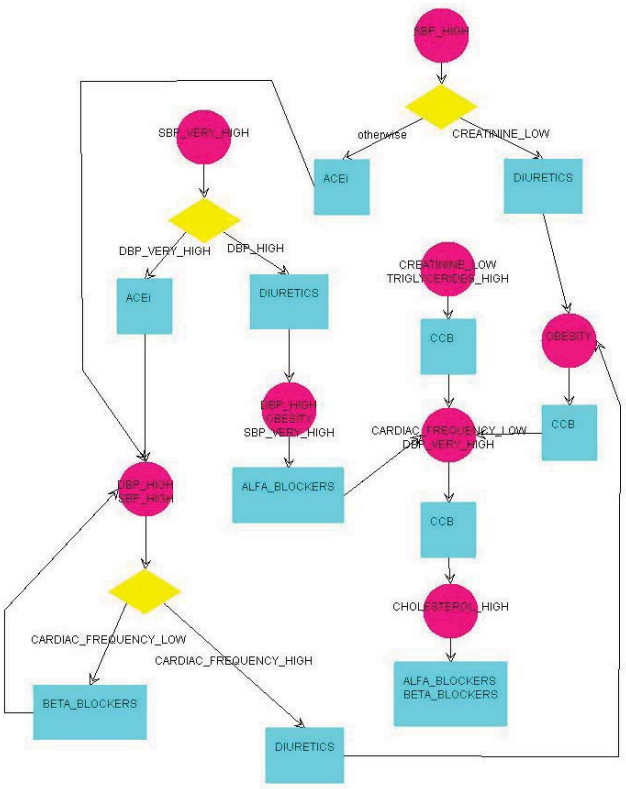

Figure 4. CA of pharmacological treatment of hypertension

Table 1. Ratios of classification of encounters in the experiments

	First CA		Second CA	
Number of encounters	131	100%	38	100%
Classified	109	83%	35	92%
Well classified	97	74%	19	50%

Attaining to these numbers and also to the CAs in figures 3 and 4 the main observations are:

- From a technical point of view both CAs are successful in the correct identification of the patient conditions with 83% and 92% of hits. The first CA also has a correct decision (74%) on the treatment the patients have to follow, where a pure random choice would have obtained just a 50%. In the second CA 50% of good classified encounters may be interpreted as a bad result, but pure random choice is only 14% and we must recall that the training sample is quite short to represent the treatment variability.
- From a medical perspective, two medical doctors have been asked to evaluate the CAs. Both agreed that the first CA describes correct medical actuations, except that the algorithm was very reluctant to prescribe drugs when the patient has SBP high. Both doctors recommend drugs in such cases. The expert analysis of the second CA is less positive, mainly because the state variables in the test were not all of the variables that should be used. This fact is also in close relation to the low good classified percentage discussed in the previous observation. In this CA, doctors also detected medical incoherences, which drove us to identify encounters reflecting odd medical actuations.
- From an intuitive analysis, the first CA shows interesting subtreatments as, for example, the stages of treatment when the SBP and the DBP are very high. In this case the treatment of the patient is centered in the reduction of DBP and then the SBP. Some other interesting facts are that, in case of bad lifestyle or obesity, the patient is recommended to change his lifestyle and then he is treated of a feasible high SBP.

5 CONCLUSION

This work shows that it is possible to obtain SA and SAP structures from data and to connect these structures in a CA. The quality of the results depends of the selection of the state and the action variables and also of the quality and the quantity of medical data available.

For the same dataset, it is possible to obtain alternative CAs each one representing a different point of view of the treatment, depending on the state and action variables selected.

Another issue is the quality of the medical data. Using data that is not representative of all the variability of the medical treatment we are trying to learn, may cause that the final CA is not medically correct. Sometimes, this fact may be used to detect the particularities of the treatments in a hospital or health-care irregularities.

In the studied cases, some problems as *Obesity* may require the application of actions that go beyond the following encounter or patient state. These long-term prognosis is not detected with the explained methodology.

The methodology has been tested at a structural level, but a further refinement is still required before the algorithm could be accepted by physicians or used in clinical practice.

ACKNOWLEDGEMENTS

This research has been partially funded by the EU project K4CARE (IST-2004-026968) and the Spanish National project HIGIA (TIN2006-15453).

REFERENCES

[1] National guideline clearninghouse, http://www.guideline.gov/.
[2] *Clinical Practice Guidelines: Directions for a New Program*, eds., M. J. Field and K.N. Lohr, Institute of Medicine, Washington, DC: National Academy Press, 1990.
[3] Gersende Georg, *Computerization of Clinical Guidelines: an Application of Medical Document Processing*, volume 185, 1–30, Springer-Verlag Berlin Heidelberg, 2005.
[4] D.C. Hadorn, *Use of algorithms in clinical guideline development*, 93–104, number 95-0009 in Clinical Practice Guideline Development: Methodology Perspectives, Agency for Health Care Policy and Research, Rockville, MD, 1995.
[5] F. Real and D. Riaño, *Automatic combination of Formal Intervention Plans using SDA* representation model*, volume LNAI 4924, 75–86, Springer, 2007.
[6] F. Real, D. Riaño, and J.A. Bohada, 'Automatic generation of formal intervention plans based on the sda representation model', in *CBMS*, pp. 575–580, (2007).
[7] D. Riaño, 'The SDA model: A set theory approach', in *CBMS*, pp. 563–568, (2007).
[8] Ravi Shankar, Samson Tu, and Mark Musen, 'Use of protege-2000 to encode clinical guidelines', Technical report, Stanford Medical Informatics Reports, (2002).
[9] S. Shayegani, *Knowledge Modeling to Develop a Clinical Practice Guideline Ontology: Towards Computerization and Merging of Clinical Practice Guidelines*, Ph.D. dissertation, Dalhousie University, Halifax, Nova Scotia, Canada, 2007.