

Inference of Markov Chain Models by Using k-Testable Language: Application on Aging People

Catherine Combes¹(✉) and Jean Azema²

¹ University of Lyon, Hubert CURIEN Laboratory,
UMR CNRS 5516, University of Jean Monnet, 18 rue Benoît Lauras,
42023 Saint-Etienne Cedex 2, France
combes@univ-st-etienne.fr

² University of Jean Monnet, 23 avenue du Docteur Paul Michelon,
42023 Saint-Etienne Cedex 2, France
azema@univ-st-etienne.fr

Abstract. We investigate the contribution of unsupervised learning and regular grammatical inference to respectively identify profiles of elderly people and their development over time in order to evaluate care needs (human, financial and physical resources). Grammatical Inference (also known as automata induction, grammar induction and automatic language acquisition) allows grammar and language learning from data. Machine learning by using grammar has a variety of applications: pattern recognition, adaptive intelligent agents, diagnosis, biology, systems modelling, prediction, natural language acquisition, data mining... The proposed approach is based on regular grammar. An adaptation of k-Testable Languages in the Strict Sense Inference algorithm is proposed in order to infer a probabilistic automaton from which a Markovian model which has a discrete (finite or countable) state-space has been deduced. In simulating the corresponding Markov chain model, it is possible to obtain information on population ageing. We have verified if our observed system conforms to a unique long term state vector, called the stationary distribution and the steady-state.

Keywords: Grammar inference · k-Testable language in strict sense · Probabilistic deterministic finite automata · Timed-transition systems · Evolution of elderly people disability

1 Introduction

Demographic shifts in the population and the fact that people are living longer have created an awareness that the health care system is and will be increasingly difficult to control, organize and finance especially where the ageing population are concerned. The senior citizen population is increasing along with the diversity of their health backgrounds and medico-social needs which cannot be provided easily because of health aspects, social conventions and lifestyles that are intertwined with the ageing process. Long-term care is a variety of services that includes medical and non-medical

care to people who have a chronic illness or disability. This illness or disability could include a problem with memory loss, confusion, or disorientation. This is called cognitive impairment and can result from conditions such as Alzheimer's disease [15]. Care needs often progress as age or as chronic illness or as disability progresses. Long-term care helps meet health or personal needs. Most long-term care is to assist people with support services such as activities of daily living like dressing, bathing, and using the toilet. Approximately 70 % of individuals over the age of 65 will require at least some type of long-term care services during their lifetime. Over 40 % will need care in a nursing home for some period of time. Nursing homes provide long-term care to people who need more extensive care, particularly those whose needs include nursing care or 24-hour supervision in addition to their personal care needs. We focus our interest on nursing homes.

This presentation is split up into eight sections. After an introduction describing the scope of the study, the study context is presented in Sect. 2. The characteristics of the collected data are described in Sect. 3. The profiles of residents obtained by using cluster analysis are presented in Sect. 4. A brief review of previous works is presented in Sect. 5. Section 6 deals with the techniques used (regular probabilistic grammar inference) to model the automaton symbolizing the changing profiles and their development over time. Starting from this automaton, a Markov model is deduced. Thereby, it is possible to verify if our system is achieving a steady state. Section 7 presents the results obtained concerning the four medical nursing homes (called Bernadette, Soleil, Les Myosotis, Val Dorlay situated in France) and dementia. We conclude with some future works.

2 Study Context

This project is being carried out in close collaboration with a French mutual benefit organization called "Mutualité Française de la Loire" which manages several nursing homes.

To fully explore the opportunities for our approach, we propose a modelling environment. This allows one to manage the resources, to elaborate medico-social resource planning and to simulate them in order to evaluate the performance of each of them (Fig. 1).

The Decisional Information System is crucial in the methodological framework. The objective is to design an environment for decision aid-tools dedicated to health-care and social services professionals for strategic and tactical decisions such that:

- Which activity should be developed, and at what cost?
- Whether or not to open additional facilities - and if so which ones?
- etc.

The aim is to provide a certain standard of care to dependent old people in nursing homes, through the sharing of resources (staff, finance and equipment) thanks to all the key indicators (via the managers' choices).

The methodological framework is described in Fig. 2.

So, in order to do this, we have to identify different senior citizen profiles as well as the length of stay according to these profiles and to study their development over time. The collected data comes from administrative data, previous medical history, etc.

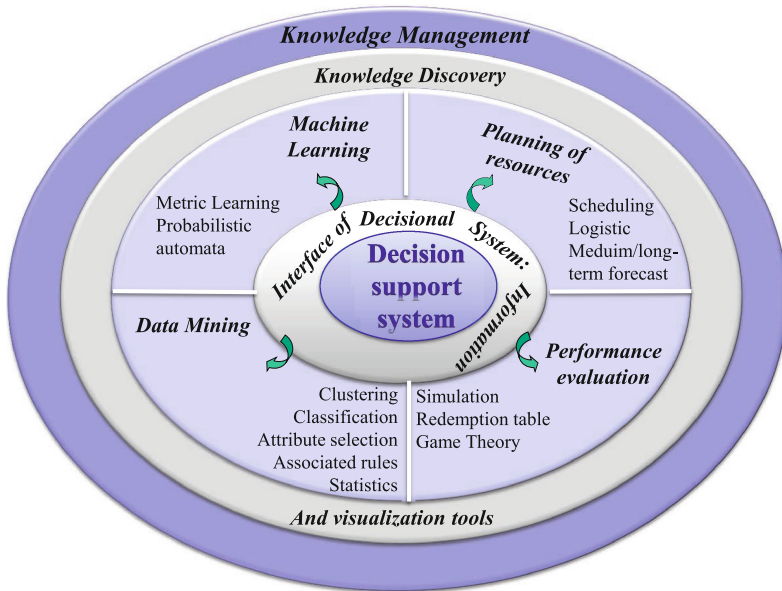


Fig. 1. Modelling environment

We evaluate the workload in function of the profile of each elderly person in the nursing home. Based on nursing home pricing (for accommodation and care in function of the degree of dependence...), we can verify that:

- the budget will not be exceeded,
- we have enough staff to carry out the workload.

Simulations are used to forecast population ageing. So we can identify the workload in the short-term, medium-term and long-term and estimate the resources needed. From the potential elderly people needs, we can also evaluate which profiles should be accepted to ensure that both quality and safety criteria are respected from a workload and financial point of view.

In this article, we will present the study of the progression over time of elderly people autonomy-disability. The steps of the project consist in:

1. The specification of elderly people profiles by using unsupervised learning approach [9],
2. The study of the development of these profiles over time by using a probabilistic graph of transitions between the clusters inferred by k-TSSI (k-Testable Languages in the Strict Sense Inference) algorithm. The objective is to deduce Markov process which has a discrete (finite or countable) state-space.
3. Discrete-time Markov chain simulation is used to forecast population ageing. It allows to identify the elderly people care needs and the workload in short-term, medium-term and long-term and to predict the future costs. An application is presented in [8].

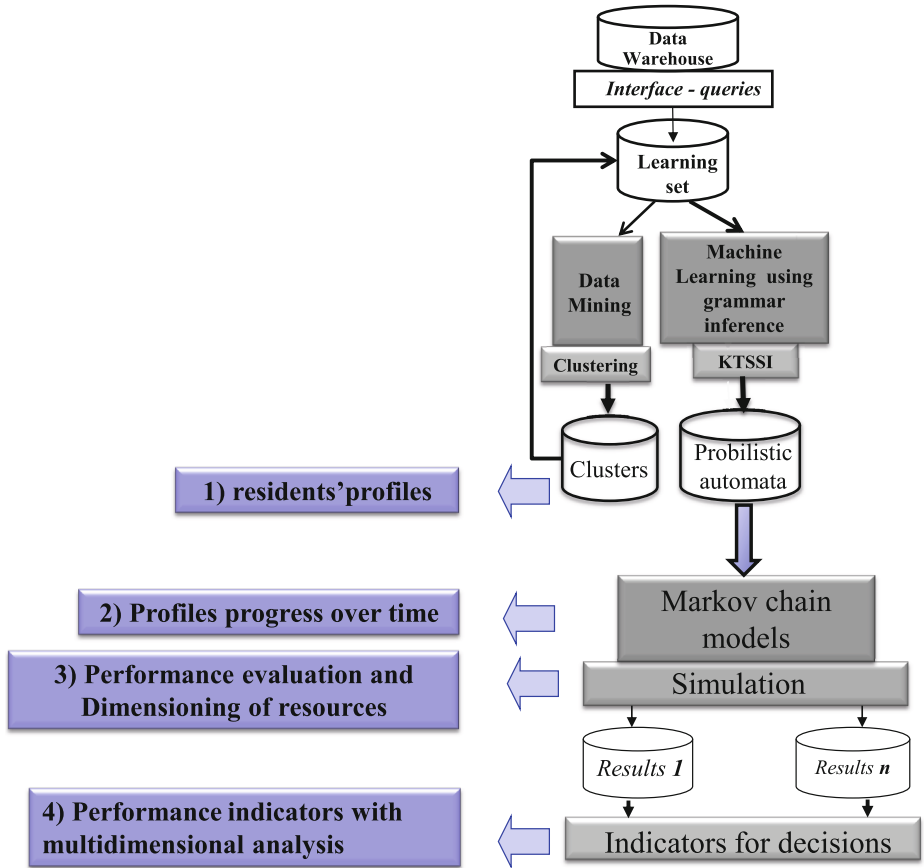


Fig. 2. Methodological framework

3 Data Collected

The quantitative data arises from the databases and the corresponding information system deals with the evaluation of autonomy/disability of elderly people. Dependence evaluation in France is carried out using a specific national grid called AGGIR: Autonomy-Gerontology-Group-Iso-Resources. The quantitative data concerns 628 residents and more than 2,200 observations of independence evaluations. The evaluations are made by the resident doctor in collaboration with the medical staff. An item can be evaluated using the four adverbs (see Fig. 3):

- Spontaneously corresponding to the letter *S*,
- Entirely corresponding to the letter *E*,
- Correctly corresponding to the letter *C*,
- Usually corresponding to the letter *U*.

The codification is the following. If all four adverbs are marked, the code is **C**. If less than four adverbs are checked (three or two or one), the code is **B**. If no adverb is checked, the code is **A**.

The proposed algorithm uses numerical data. So, the corresponding values are:

- 0 for code **A** meaning the person can do it alone,
- 1 for code **B** meaning the person can partially do it,
- 2 for code **C** meaning the person cannot do it alone.

The first step is to analyze the degree of autonomy-disability in order to identify clusters.

The screenshot shows the A.G.G.I.R. grid interface. At the top, there are input fields for Patient number, Date of birth, Age, and Evaluation date (20/09/2012). Below these is a dropdown menu for 'Patient in nursing home'. The main grid is titled 'A.G.G.I.R.' and has columns for 'Not done', 'S', 'E', 'C', 'U', and 'No adverb'. The rows represent different activities: Transferring, Moving indoors, Washing (Upper body, Lower body), Toilet usage (Urinary, Faecal), Dressing (Upper body, Middle body, Lower body), Food (To serve, To eat), Orientation (In time, In space), and Coherence (Communication behavior). To the right of the grid is a vertical column of codes (A, B, C) corresponding to each row. A legend on the right explains the codes: A: no adverb is checked; B: some adverbs are checked; C: spontaneously, usually, entirely and correctly (the whole set of adverbs are checked). The legend also includes a red box around the code 'C' and a blue box around the code 'B'. Below the legend is a section for 'Iso-Resource Group' with an empty box.

Fig. 3. A.G.G.I.R. grid

4 Identification of Residents' Profiles

The aim is to find feature-patterns related to the autonomy-disability level of elderly people living in nursing homes. These levels correspond to profiles based on the people's ability to perform activities of daily living like being able to wash, dress and move. To achieve this aim, an unsupervised learning approach is proposed [9]. It is based on principal component analysis technique to direct the determination of the clusters with self-organizing partitions. Cluster analysis is made on the 8 variables: Transferring to or from bed or chair, Moving indoors, Washing, Toilet, Dressing, Food, Orientation, Coherence. The cluster analysis identifies two kinds of patterns (see Fig. 4):

- The decline in executive functions regarding to motor and functional abilities called apraxia disorders,
- The cognitive impairment and neuropsychological deficits.

By combining clustering with a machine learning process, we could be able to predict the development of physical autonomy loss or mental autonomy loss in elderly people over time. To reach this objective, we use machine learning approach based on grammar inference in order to infer a probabilistic automaton. In the article, we only present the patients' profiles evolution regarding to upper body functional disorders (cognitive impairment).

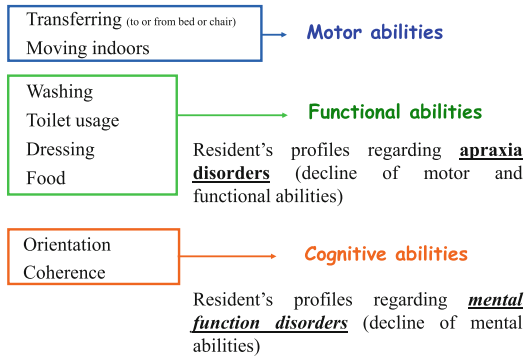


Fig. 4. Contribution of the clustering

5 Related Works

We want to obtain a probabilistic graph of transitions between states (clusters) with the length-of-stay in each state (temporal state representations). It is also interesting to study cluster succession of length k (for example, the 3 last states of resident's clusters). Probabilistic automata are used in various areas in pattern recognition or in fields to which pattern recognition is linked. Different concept learning algorithms have been developed for different types of concepts.

We are interested by the class of regular grammars that are the simplest class of formal grammars in the Chomsky hierarchy and it consists in the identification of the corresponding learning of deterministic finite automata (DFA).

The learning of DFA, also called regular inference is based on acceptance of regular languages which allow to model the behaviour of systems. The aim consists in constructing a DFA from information about the set of words it accepts. There are many algorithms for learning DFAs, the most well-known being the algorithm due to Dana Angluin [4, 5]. There are many approaches for regular inference [6, 7, 11, 16, 17, 19]. For more information, the book [14] presents an overview on learning automata and grammar inference.

A finite automaton with transition probabilities represents a distribution over the set of all strings defined over a finite alphabet. The articles [18, 24] present a survey and a study of the relations and properties of probabilistic finite-automata and tree. The article [10] clarifies the links between probabilistic automata and Hidden Markov Models (HMM). In a first part of this work, the authors present:

- the probabilities distributions generated by these models,
- the necessary and sufficient conditions for an automaton to define a probabilistic language.

The authors show that on the one hand, probabilistic deterministic finite automata (PDFA) form a proper subclass of probabilistic non-deterministic automata (PNFA) and on the other hand, PNFA and HMM are equivalent.

However, there are almost no extensions of these algorithms to the setting of timed systems. A variant of a DFA including the notion of time is called Timed Automaton (TA). These models are based on the fact that each symbol of a word occurs at a certain point in time. TA can see that an automaton that generates strings with event-time value pairs called timed strings. Very few works exist in the domain [1–3, 13, 20, 21]. Timed automata correspond to finite state models where explicit notion of time is taken into account and is represented by timed events. Time can be modelled in different ways, e.g. discrete or continuous. The more recent works [22, 23] propose an algorithm for learning simple timed automata, known as Deterministic Real-Time Automata (DRTA) where the transitions of real-time automata can have a temporal constraint on the time of occurrence of the current symbol relative to the previous symbol. The main drawback of such approaches is that it can result in an exponential blow-up of both the input data and the resulting size of the model and the obtained models are quite difficult to interpret.

The main objective of the study is the ability to analyze an identified model in identifying automaton models from observations. We want to take an established method to learn a DFA and apply it to our timed sequences. Our problem could be modelled as a timed-state transition graph, a probabilistic deterministic finite automaton (PDFA) taking into account timed-event. We also have a set of positive timed-strings (or time-stamped event sequences).

From a set of labelled (positive only) time stamped event sequences, the problem to solve is to find the automaton model that most likely produce the data. We do not want to learn (identify) a DRTA such that [22], with time constraints because we do not exactly have the same problematic of real-time system. We only have timed-strings from which we propose to automatically deduce a Markov chain model.

The k -TSSI (k -Testable Languages in the Strict Sense Inference) algorithm [11, 12] could be useful, convenient and suitable for two reasons: the simplicity of implementation and the possibility to take into account memory effects (macro-states). The inductive inference of the class of k -testable languages in the strict sense (k -TLSS) has been studied and adapted to local languages, N -grams and tree languages. A k -TLSS is essentially defined by a finite set of substrings of length k that are permitted to appear in the strings of then language. Given a size k of memory, the objective is to find an automaton for the language. This subclass of language called k -testable language has the property that the next character is only dependent on the previous $k-1$ characters. In our case, it is interesting to be able to identify the substrings (memory) of length k .

But, our goal is to infer a timed-system model and an automaton inferred by the k -TSSI algorithm does not take into account the timed strings.

The problem is also that it is difficult to take into account a set of substrings of length $k(k > 1)$ and the algorithm is not generalized to probabilistic timed-automata. In

this section we propose a model in order to take into account the concept of time in the automaton inferred by the k-TSSI algorithm (i.e. the duration of time a resident spends in a particular cluster) taking into account timed-transition system. In the next section, we present the implementation of the model.

6 Development of Patients' Profiles: Model Implementation

We present an algorithm for inferring a model of a timed transition system based on the K-Testable Language in Strict Sense (K-TLSS).

The method consists of:

1. Learning a Deterministic Finite Automata (DFA) of timed-transition systems by using an extension of k-TSSI algorithm.
2. Transforming this DFA into a probabilistic DFA.
3. Converting this probabilistic DFA in a Markov chain model.

6.1 Preliminaries

The aim of grammatical inference is to learn models of languages from sample sentences in these languages. A sentence can be any structured composition of primitive elements or symbols, though the most common type of composition is concatenation. So we infer grammar and the corresponding representation is an automaton.

A finite automaton is a 5-tuple $A = (\mathbf{Q}, \Sigma, \delta, \mathbf{q}_0, \mathbf{F})$ with:

- Σ : a finite input alphabet of symbols,
- \mathbf{Q} : a finite set of states with \mathbf{q}_0 as start state,
- \mathbf{F} : a set of final states ($\mathbf{F} \subseteq \mathbf{Q}$),
- δ : a transition function of $\mathbf{Q} \times \Sigma \rightarrow \mathbf{Q}$. So that $q' = \delta(q, \sigma)$ returns a state for current state q and input symbol σ from Σ . Each transition is noted by 3-tuple (q, σ, q') .

If for all $q \in \mathbf{Q}$ and for all $\sigma \in \Sigma$, $\delta(q, \sigma)$ corresponds to a unique state of \mathbf{Q} , then the automaton is said to be a Deterministic Finite Automaton (DFA). Grammatical inference refers to the process of learning rules from a set of labelled examples. It belongs to a class of inductive inference problems [4] in which the target domain is a formal language (a set of strings generated from some alphabet Σ) and the hypothesis space is a family of grammars. It is also often referred to as automata induction, grammar induction, or automatic language acquisition. The inference process aims at finding a minimum automaton (the canonical automaton) that is compatible with the examples. In regular grammar inference, we have a finite alphabet Σ and a regular language $L \subseteq \Sigma^*$ where:

- Σ^* : the set of all finite length strings generated from Σ ,
- L : a sub-set of Σ^* corresponding to the words recognized from the automaton A .

Given a set of examples that are in the language (I_+) and a (possibly empty) set of examples not in the language (I_-), the task is to infer a deterministic finite automaton A that accepts the examples in I_+ and rejects the examples in I_- .

6.2 Automata Induction

The k-TSSI algorithm [11] allows us to infer k-Testable Languages in the Strict Sense. The inductive inference of the class of k-Testable Languages in the Strict Sense is defined by a finite set of substrings of length k that are allowed to appear in the strings of the language. Given a positive sample $I_+ \in \mathbf{L}$ of strings of an unknown language, a deterministic finite-state automaton that recognizes the smallest k-TLSS containing I_+ is obtained. An automaton inferred by the k-TSSI algorithm is by its construction, non-ambiguous. Moreover, our choice is justified by the fact that k-testable ($k > 1$) can take into account a memory effect (i.e. N-gram). Indeed, we observed during data analysis that the change in evolution of the autonomy/disability state depends on the previous resident's states and their diseases (especially for chronic and disabling diseases such as osteoarticular degenerative diseases, anxio-depressive disorder, behavioural disorders...). To illustrate our approach and for the sake of simplicity, we will present in this article, the results obtained with 1-TSSL (the next state depends only on the previous states) in order to explain how we turn the timed transitions into sequences (time-series). We choose to divide up the length-of-stay in each cluster (for example, one discrete step $T = 30$ days).

Consequently, the corresponding automaton is a 6-tuple $(Q, \Sigma, \delta, q_0, F, d)$ where d corresponds to the length-of-stay in the clusters. At each cluster, we assign a symbol $\sigma \in \Sigma$. Each symbol is represented by a pair (σ_i, d_i) , where d_i is the delay on the symbol σ_i which corresponds to an application $d: \Sigma \rightarrow \mathbf{N}$ with $d(\sigma_i) = d_i$.

In the following sections, we explain the implementation of the model through an example (on only six residents: 7, 12, 17, 14, 8, 44 corresponding to an excerpt of the collected data).

Setting Up the Alphabet. The assessment of elderly people's autonomy/disability allows us to classify residents into five levels of mental dependence situation (5 to 1 in decreasing order of severity). Figure 5 presents the data collected from the database.

The resident assessment is made on different dates. For example, resident number 7 was evaluated at level 3 (mental disorder) on the 06/24/2002. For all the assessments concerning resident number 7, we can deduce the sequence: **3321111**. But this sequence does not express the amount of time the person spends in each state (level of mental disorder). In this model, each symbol of a word occurs at a certain point in time.

From such observations, we only obtain positive data I_+ , and now we have to describe how we can obtain timed strings. The following paragraph present the basic notions explaining how we take into account timed-transitions between clusters.

Preliminary Mapping of the Set of Strings. The objective is to obtain a stochastic state transition graph taking into account the length-of-stay in each state. So we have to associate for each occurrence of a symbol (event) in order to model time value. In practice, we use the evaluation date.

The **first step** consists in the definition of the alphabet (the set Σ). The set Σ is based on an alphabet of 6 symbols - $\{a, b, c, d, e, f\}$ which correspond to:

- a length-of-stay in cluster number 1 during a given period T (example: 30 days) meaning $d(a) = T$,
- b length-of-stay in cluster number 2 during a given period,
- etc. (until the symbol e for cluster number 5).

Problem:

How can we analyze the resident's level-score of mental autonomy-disability and identify the development over time?

Objective:

To obtain stochastic state transition graph taking into account the length-of-stay in each state (5 to 1).

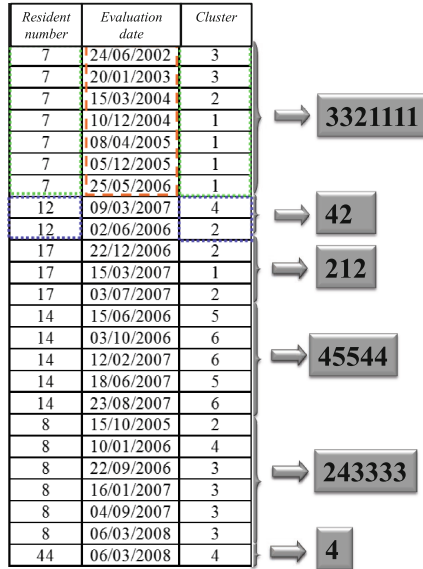


Fig. 5. Data and sequencing

The symbol f models the fact that a resident can leave the nursing home or corresponds to the last resident assessment during the last 30 days before the data extraction. It is only used when we want to deduce the Markov model. Consequently, in the following example, the symbol f does not exist in Fig. 6.

The **second step** concerns the identification of the words which corresponds to the translation of the initial sequence in order to take into account length of time spent in each cluster. Resident number 7 stayed in cluster number 3 from 06/24/2002 to 03/15/2004 (date at which the resident was evaluated and changed to cluster number 2). Thus resident number 7 stayed in cluster number 3 for about 22 periods of 30 days. The symbol modeling cluster number 3 for 30 days is c , consequently the initial sequence “33” becomes “cccccccccccccccccccc”. The resident stayed in cluster number 2 for 9 periods...

And the corresponding word is:

ccccccccccccccccccccbbbbbbbaaaaaaaaaaaaaaaaaa

So we obtain the set $I_+ \subseteq \Sigma^*$. I_+ corresponds to the learning set from which the automaton is inferred. The initial set of sequences (Fig. 5)

{3321111, 42, 212, 56656, 243333, 4}

becomes:

$I_+ = \{ccccccccccccccccccccbbbbbbbaaaaaaaaaaaaaaaaaa, ddb, bbbbaaab, ddddeeeeeeddee, bbbdddddcccccccccccccc, dd\}$

From the set I_+ by using k-TSSI algorithm (to simplify, we present the case corresponding to $k = 1$), we obtain the automaton described in Fig. 6. The algorithm [12]

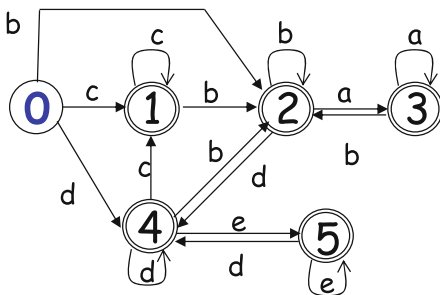


Fig. 6. The automaton inferred by the algorithm k-TSSI with $q_0 = 0$

consists in building the sets \mathbf{Q} , Σ , δ , q_0 , \mathbf{F} by observation of the corresponding events in the training strings. From these sets, a finite-state automaton that recognizes the associated language is straightforwardly built.

Let I_+ be a positive sample of the regular language L .

Input

k // substrings of length k

$I_+ = \{x_1, \dots, x_{|I_+|}\}$ // collected sample

$D = \{(\sigma, d(\sigma)) \text{ with } \sigma \in \Sigma\}$

Output $A_k = (\mathbf{Q}, \Sigma, \delta, q_0, \mathbf{F}, d)$ // the obtained automaton

Begin

$Q \leftarrow q_0$ // initial state

$\Sigma \leftarrow \emptyset, \delta \leftarrow \emptyset, \mathbf{F} \leftarrow \emptyset$

$q_0 \leftarrow \lambda$ // λ corresponds to an empty string

For $i=1$ **until** $|I_+|$ // for all words x_i in I_+

$q' \leftarrow q_0$

For $j=1$ **until** $|x_i|$ // For all symbol x_{ij} of the word $x_i \in I_+$

$\Sigma \leftarrow \Sigma \cup \{x_{ij}\}$ // to add a symbol x_{ij} to the alphabet

$y \leftarrow q' x_{ij}$ // to build the substrings of length $\leq k$ regarding the word x_i

If $|y| > k$ **then**

$y \leftarrow y_2, \dots, y | y |$ // the length of the substring must be equal to k .

// => Shift of one symbol

EndIf

$q \leftarrow y$ // the state representing the substring

$Q \leftarrow Q \cup \{q\}$ // to add this state

$\delta \leftarrow \delta \cup \{(q', x_{ij}, q)\}$ // to add the transition

If $j = |x_i|$ **Then** // at the end of the word

$\mathbf{F} \leftarrow \mathbf{F} \cup \{q\}$ // the state is final

EndIf

$q' \leftarrow q$ // progression on the state in order to process the next symbol

EndFor

EndFor

Return A_k

End k-TSSI

6.3 Computation of Probabilities

The automaton is inferred by the k-TSSI algorithm. We have to associate transition probabilities with states. In order to compute these probabilities, we use the learning set I_+ . From the words of set I_+ , when they are recognized by the automaton inferred by k-TSSI, we count:

- The transition between two states for a given symbol (transition from the state q by the symbol σ): $cp_{\delta(q,\sigma)}$,
- each transition in a state q : cp_q ,
- if a state q is the final state (end of the words): cp_{q_final} .

For the algorithm, we use the three epochs-counts in order to estimate the probabilities. The algorithm computing the probabilities from a learning set is the following.

```

Input  $I_+ = \{x_1, \dots, x_{|I_+|}\}$  //collected sample
         $A_k = (\mathbf{Q}, \mathbf{\Sigma}, \mathbf{\delta}, \mathbf{q}_0, \mathbf{F}, \mathbf{d})$  //the inferred
        automaton
Output  $PA_k = \{p_{\delta(q,x_{ij})}, p_{q\_final}\}$  //the obtained probabilities
Begin
For  $i=1$  until  $|I_+|$  //for all words  $x_i$  in  $I_+$ 
     $q \leftarrow q_0$ 
    For  $j=1$  until  $|x_i|$  //for all symbol  $x_{ij}$  of the word  $x_i \in I_+$ 
         $q' \leftarrow \delta(q, x_{ij})$  //the corresponding transition
         $cp_q ++$  //epoch-count in passing state
         $cp_{\delta(q,x_{ij})} ++$  //epoch-count in passing transition
         $q \leftarrow q'$ 
    EndFor
     $cp_{q\_final} ++$  //epoch-count concerning the final states
     $cp_q ++$ 
EndFor
For all  $q \in \mathbf{Q}$ 
     $p_{q\_final} = cp_{q\_final} / cp_q$  //Computation of final-state probabilities
EndFor
For all  $\delta(q, \sigma) \in \mathbf{\delta}$ 
     $p_{\delta(q,\sigma)} = cp_{\delta(q,\sigma)} / cp_q$  //Computation of transition probabilities
EndFor
Return  $PA_k$ 

```

From the automaton (Fig. 6) and the set I_+ , we count:

- cp_q : The number of times the state q was used while generating the set of words of I_+ .
- $cp_{\delta(q,\sigma)}$: The number of times the transition $\delta(q,\sigma)$ was used while generating the set of words of I_+ .
- cp_{q_final} : The number of times that a state q is final (state gets at end of the words).

The obtained results from the sample presented in Fig. 5 are:

- $cp_q = (6_{(0)}, 39_{(1)}, 19_{(2)}, 22_{(3)}, 17_{(4)}, 10_{(5)})$,
- $cp_{q_final} = (0_{(0)}, 1_{(1)}, 2_{(2)}, 1_{(3)}, 1_{(4)}, 1_{(5)})$,
- $cp_{\delta(q,\sigma)} = (2_{\delta(0,b)}, 1_{\delta(0,c)}, 3_{\delta(0,d)}, 0_{\delta(0,e)}, 1_{\delta(1,b)}, 37_{\delta(1,c)}, 2_{\delta(2,a)}, 14_{\delta(2,b)}, 1_{\delta(2,d)}, 20_{\delta(3,a)}, 1_{\delta(3,b)}, 1_{\delta(4,b)}, 1_{\delta(4,c)}, 12_{\delta(4,d)}, 2_{\delta(4,e)}, 1_{\delta(5,d)}, 8_{\delta(5,e)})$.

And afterwards, we deduce *the probabilities*:

- $P_{q_final} = cp_{q_final} / cp_q$
 $= (0/6_{(0)}, 1/39_{(1)}, 2/19_{(2)}, 1/22_{(3)}, 1/17_{(4)}, 1/10_{(5)})$,
- $P_{\delta(q,\sigma)} = cp_{\delta(q,\sigma)} / cp_q$
 $= (2/6_{\delta(0,b)}, 1/6_{\delta(0,c)}, 3/6_{\delta(0,d)}, 1/39_{\delta(1,b)}, 37/39_{\delta(1,c)}, 2/19_{\delta(2,a)}, 14/19_{\delta(2,b)}, 1/19_{\delta(2,d)}, 20/22_{\delta(3,a)}, 1/22_{\delta(3,b)}, 1/17_{\delta(4,b)}, 1/17_{\delta(4,c)}, 12/17_{\delta(4,d)}, 2/17_{\delta(4,e)}, 1/10_{\delta(5,d)}, 8/10_{\delta(5,e)})$.

So we obtain the probabilistic deterministic automaton where the time series are taken into account. The advantage of using 1-TSSL (k-TSSI algorithm with $k = 1$) lies in the fact that one state corresponds to one symbol. We have added a new symbol f and a final state q_6 in order to facilitate the translation of the probabilistic automaton into a Markov process. For all q states where $p_{q_final} > 0$, we add a transition $\delta(q, g) = q_6$, $P_{\delta(q,g)} = p_{q_final}$ and $p_{q_final} \leftarrow 0$. We note that $p_{q_6_final} = 1$.

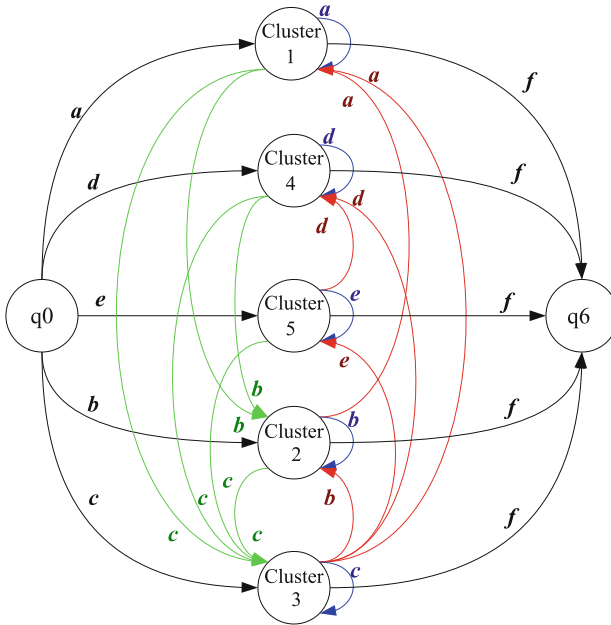


Fig. 7. The automaton inferred by the algorithm k-TSSL (Soleil nursing home: residents suffering from dementia).

Table 1. The corresponding probability matrix of transitions between states (Fig. 7).

To \Rightarrow From \Leftarrow	Cluster 5	Cluster4	Cluster3	Cluster2	Cluster1	q_6
q_0	0.5072	0.0580	0.3333	0.0290	0.0725	
Cluster5	0.9738	0.0005	0.0009	0	0	0.0248
Cluster4	0.0629	0.9021	0.0210	0	0	0.0140
Cluster3	0.0229	0.0134	0.9408	0.0019	0.0019	0.0191
Cluster2	0	0.0299	0.0299	0.8955	0	0.0448
Cluster1	0	0	0.0122	0.0488	0.9268	0.0122

From patients’ file living in Soleil nursing home and suffering from Alzheimer disease, the probability matrix of transitions between states and the corresponding automaton are respectively presented in Table 1 and in Fig. 7.

6.4 Markov Model

The final state q_6 does not only represent the resident state when they left the system but also the last resident assessment (resident present in the system at the date of database extraction).

In order to obtain the Markov chain model, we have to compute the probabilities:

- Pe_i : Input probabilities (i.e. the initial resident assessments) in each $cluster_i$ ($i = 1..5$),
- Ps_i : Output probabilities (i.e. the last resident assessments when residents leave the system) in being $cluster_i$ ($i = 1..5$) after $d(cluster_i) = 30$ days (corresponding to the equidistant discrete time described in the automaton definition in the paragraph 5.2).

We have also to modify the probabilities of staying in $cluster_i$ ($i = 1..5$), regarding if the patient is staying in the nursing home at the at the date of database extraction (these evaluations are taken into account in the transition with the symbol f to q_6 in Table 1). We add the number of evaluations in the corresponding $cluster_i$. It is the reason that the probability to be in cluster1, (initially is 0.9738 in Table 1) becomes 0.9902 in the Markov matrix.

When a resident leaves the system, he is immediately replaced by a new resident. Consequently, two other probabilities are taken into account PE and PS . The Markov matrix is presented in Table 2.

We verify if the system reaches a steady state. Out of definition, an eigenvector x is associated to eigenvalue 1 if: $A * x = 1 * x$.

(A corresponding to the probabilities matrix presented in Table 2)

If an eigenvector of x is associated to a unique eigenvalue 1 , such a vector is called a **steady state vector**. If we identify only one eigenvalue 1 , then the distribution is said to be irreducible and aperiodic.

The eigenvector associated with the eigenvalue 1 has been computed. We have one eigenvalue 1 and the corresponding eigenvector x is the following:

0.00692 0.01263 0.01966 0.12108 0.03768 0.79510 0.00693.

The interpretation of this eigenvector is that the system (ratio of the resident profiles without 0.69 % of resident turnover of input/output in the nursing home) evolves towards a state where the percentages of population are:

- 1.28 % are in *cluster1*,
- 1.99 % are in *cluster2*,
- 12.28 % are in *cluster3*,
- 3.82 % are in *cluster4*,
- 80.63 % are in *cluster5*.

Table 2. The Markov matrix obtained from the collected data - Soleil Nursing home: patient suffering from dementia.

	Pe_i	<i>Cluster1</i>	<i>Cluster2</i>	<i>Cluster3</i>	<i>Cluster4</i>	<i>Cluster5</i>	<i>PSS</i>
PEE	0	0	0	0	0	0	1
<i>Cluster1</i>	0.0725	0.9390	0	0.0019	0	0	0
<i>Cluster2</i>	0.0290	0.0488	0.9403	0.0019	0	0	0
<i>Cluster3</i>	0.3333	0.0122	0.0299	0.9580	0.0210	0.0009	0
<i>Cluster4</i>	0.0580	0	0.0299	0.0134	0.9161	0.0005	0
<i>Cluster5</i>	0.5072	0	0	0.0229	0.0629	0.9902	0
Psi	0	0	0	0.0019	0	0.0084	0

7 Experiments

The experiment deals with the evolution of a patient's loss of cognitive autonomy over time. Table 3 presents the steady state vectors from different samples which correspond to a stable condition that does not change over time or in which change in one direction is continually balanced by change in another. We see that the decline is more significant for elderly people with dementia than non-demented elderly people.

Now, we simulate the evolution over time by using transition matrix used to model the Markov chain concerning each population. The results concerning the patients' profile progress in 2 years are presented in Tables 4 and 5.

Table 3. Steady state: population staying in medical nursing homes.

	<i>4 Nursing Homes</i>	<i>Patient Without Dementia Disease</i>	<i>Patient Suffering from Dementia</i>
<i>Cluster5</i>	3.57%	35.98%	0.32%
<i>Cluster4</i>	13.42%	27.00%	1.93%
<i>Cluster3</i>	27.80%	15.96%	5.21%
<i>Cluster2</i>	11.54%	5.65%	6.84%
<i>Cluster1</i>	43.66%	15.40%	85.69%

Table 4. Evolution of patients' profiles in 2 years (patients without dementia)

<i>No Dementia</i>	<i>Cluster5</i>	<i>Cluster4</i>	<i>Cluster3</i>	<i>Cluster2</i>	<i>Cluster1</i>	<i>Exit</i>
<i>Cluster5</i>	50.9%	16.0%	5.8%	1.6%	2.4%	23.3%
<i>Cluster4</i>	3.8%	56.0%	10.6%	3.4%	4.1%	22.2%
<i>Cluster3</i>	4.3%	4.0%	25.2%	9.1%	13.8%	43.6%
<i>Cluster2</i>	0.8%	0.9%	11.4%	29.4%	29.6%	27.9%
<i>Cluster1</i>	0.1%	0.6%	0.7%	1.3%	33.1%	64.2%

Table 5. Evolution of patients' profiles in 2 years (patients suffering from dementia)

<i>Dementia</i>	<i>Cluster5</i>	<i>Cluster4</i>	<i>Cluster3</i>	<i>Cluster2</i>	<i>Cluster1</i>	<i>Exit</i>
<i>Cluster5</i>	9.7%	20.6%	27.1%	12.7%	19.4%	10.5%
<i>Cluster4</i>	0.5%	20.2%	32.4%	14.7%	20.0%	12.2%
<i>Cluster3</i>	0.6%	1.5%	21.8%	17.7%	34.1%	24.3%
<i>Cluster2</i>	0.1%	0.1%	1.9%	11.9%	31.7%	54.3%
<i>Cluster1</i>	0.2%	0.1%	1.5%	15.5%	64.8%	17.9%

If the patient does not suffer from dementia disease, if he is initially in cluster5, in 2 years, the probabilities that the patient will be staying in:

- *Cluster5* is 50.9 %,
- *Cluster4* is 16 %,
- *Cluster3* is 5.8 %...
- and leaves the system with a probability near to 23 %.

If the patient suffers of *dementia*, the probabilities that the patient which will be staying in:

- *Cluster5* is 9.7 %,
- *Cluster4* is 20.6 %,
- *Cluster3* is 27.1 %,...
- and leaves the system with a probability near to 10 %.

8 Conclusion

A real case application of grammatical inference to identify the progression of a resident's autonomy-disability over time has been presented. From profiles identified by using clustering approach [9], we propose preliminary results of an investigation where regular grammars are used for modeling the evolution of ageing over time. The finite automaton is inferred by using the k-TSSI algorithm and afterward modified in order to obtain a probabilistic graph of transitions between states (clusters) with the length-of-stay in each state. From this graph, we automatically deduce the corresponding Markov chain model. For the sake of simplicity, we only present in the article, the case where $k = 1$. It is evident that in this case, we can use a bi-gram. But we have also studied the evolution with $k = 2..n$. So, the approach allows identifying a sub-sequence of n items from sequences.

In future work, we will extend and validate the different models to other class of diseases. Approximately 1–1.5 % of the French population suffer from dementia and the causes of dementia are neurological disorders such as Alzheimer’s disease (which causes 50 %–70 % of all dementia), blood flow-related (vascular) disorders such as multi-infarct diseases, inherited disorders such as Huntington’s disease, and infections such as HIV [15]. In fact, we would like to simulate the patient’s progress in order to forecast and to analyze the need for long, medium and short-term care. This allows us to evaluate human, financial and physical resources in the future.

Acknowledgements. The authors would like to acknowledge Mr. F. Navarro (Chairman of the Board of “Mutualité Française” Rhône-Alpes - France), as well as all the staff who had the kind enough to entrust us this project, data to validate our models and who answered our numerous questions. The authors are very grateful to the reviewers for their comments which were both useful and helpful.

References

1. Alur, R., Courcoubetis, C., Dill, D.: Model-checking for real-time systems. In: Proceedings of the Fifth IEEE Symposium on Logic in Computer Science, pp. 414–425 (1990)
2. Alur, R., Courcoubetis, C., Dill, D.: Model-checking for probabilistic real-time systems. In: Albert, J.L., Monien, B., Artalejo, M.R. (eds.) Automata, Languages and Programming. LNCS, vol. 510, pp. 115–126. Springer, Heidelberg (1991)
3. Alur, R., Dill, D.: A theory of timed automata. *Theoret. Comput. Sci.* **126**, 183–235 (1994)
4. Angluin, D., Smith, C.H.: Inductive inference: theory and methods. *ACM Comput. Surv.* **15** (3), 237–269 (1983)
5. Angluin, D.: Learning regular sets from queries and counterexamples. *Inf. Comput.* **75**, 87–106 (1987)
6. Balczar, J.L., Daz, J., Gavald, R.: Algorithms for learning finite automata from queries: a unified view. In: Du, D.-Z., Ko, K.-I. (eds.) *Advances in Algorithms, Languages, and Complexity*, pp. 53–72. Kluwer Academic Publishers, Dordrecht (1997)
7. Bugalho, M., Oliveira, A.: Inference of regular languages using state merging algorithms with search. *Pattern Recogn.* **38**(9), 1457–1467 (2005)
8. Combes, C., Azéma, J., Dussauchoy, A.: Coupling Markov model – optimization: an application in medico-social care. In: 7e International Conference MOSIM’08, Paris, France, 31 March–2 April, pp. 1310–1319 (2008)
9. Combes, C., Azéma, J.: Clustering using principal component analysis applied to autonomy-disability of elderly people. *Decis. Support Syst.* **55**, 578–586 (2013). <http://dx.doi.org/10.1016/j.dss.2012.10.016>
10. Dupont, P., Denis, F., Esposito, Y.: Links between probabilistic automata and hidden Markov models: probability distributions, learning models and induction algorithms. *Patterns Recogn.* **38**(9), 1349–1371 (2005)
11. Garcia, P., Vidal, E.: Inference of k-testable languages in the strict sense and applications to syntactic pattern recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **12**(9), 920–925 (1990)
12. Garcia, P., Vidal, E., Oncina, J.: Learning locally testable language in strict sense. In: *Proceedings of the Workshop on Algorithmic Learning Theory*, by Japanese Society for Artificial Intelligence (1990). <http://users.dsic.upv.es/grupos/tlcc/papers/fullpapers/GVO90.pdf>

13. Grinchtein, O., Jonsson, B., Leucker, M.: Inference of timed transition systems. Proc. Int. Workshop Verification Infinite State Syst. Electron. Notes Theoret. Comput. Sci. **138**(3), 87–99 (2005)
14. de la Higuera, C.: Grammatical Inference: Learning Automata and Grammars. Cambridge University Press, New York (2010). ISBN 978-0-521-7636-5
15. Khachaturian, Z.S.: A chapter in the development of Alzheimer’s disease research: a case study of public policies on the development and funding of research programs. Alzheimer’s Dement. J. Alzheimer’s Assoc. **3**(3), 243–258 (2007)
16. Parekh, R., Honavar, V.: Learning DFA from simple examples. Mach. Learn. **44**(1/2), 9–35 (2001)
17. Parekh, R., Nichitiu, C.M., Honavar, V.: A polynomial time incremental algorithm for learning DFA. In: Proceeding of International Colloquium on Grammatical Inference: Algorithms and Applications, pp. 37–49 (1998)
18. Rico-Juan, J.R., Calera-Rubio, J., Carrasco, R.C.: Probabilistic k -testable tree languages. In: Oliveira, A.L. (ed.) ICGI 2000. LNCS (LNAI), vol. 1891, pp. 221–228. Springer, Heidelberg (2000)
19. Rivest, R.L., Schapire, R.E.: Inference of finite automata using homing sequences. Inf. Comput. **103**, 299–347 (1993)
20. Verwer, S., de Weerd M., Witteveen, C.: An algorithm for learning real-time automata. In: Adriaans, P., van Someren, M., Katrenko, S. (eds.) Proceedings of the 18th Benelearn (2007)
21. Verwer, S., de Weerd M., Witteveen, C.: One-clock deterministic timed automata are efficiently identifiable in the limit. In: Dediu, A.H., Ionescu, A.M., Martín-Vide, C. (eds.) LATA 2009. LNCS, vol. 5457, pp. 740–751. Springer, Heidelberg (2009)
22. Verwer, S., de Weerd M., Witteveen, C.: a likelihood-ratio test for identifying probabilistic deterministic real-time automata from positive data. In: Sempere, J.M., García, P. (eds.) ICGI 2010. LNCS, vol. 6339, pp. 203–216. Springer, Heidelberg (2010)
23. Verwer, S., de Weerd M., Witteveen, C.: The efficiency of identifying timed automata and the power of clocks. Inf. Comput. **209**(3), 606–625 (2011)
24. Vidal, E., Thollard, F., de la Higuera, C., Casacuberta, F., Carrasco, R.C.: Probabilistic finite-states machine. IEEE Trans. Patterns Anal. Mach. Intell. **27**(7), 1013–1039 (2005)