Mobile Data Modeling in Human Body “Network”

Bell’s Palsy Case Study

Hanen Bouali
Department of Computer Science
ISG-University of Tunis, Tunisia
41, Rue de la Liberté, Cité Bouchoucha Le Bardo 2000
Tunis- Tunisia
Hanene.bouali@gmail.com

Jalel Akaichi
Department of Computer Science
ISG-University of Tunis, Tunisia
41, Rue de la Liberté, Cité Bouchoucha Le Bardo 2000
Tunis- Tunisia
J.akaichi@gmail.com

Abstract —**The mysterious Path physiology of Bell's palsy sometimes prevents doctors to understand this disease. A fine diagnostic and an enhanced recovery surveillance are crucial for physicians to deeply be aware of the disease mechanisms. Moreover, the analyses of the patients' states may lead to the proposition of new strategies to cure the illness. This paper attempts to supervise the patients’ state evolution through the modeling of the facial nerve stream as a moving object circulating into the facial nerve “network”. Its progression through this latter, gives indication about patients’ recovery advancement and the disease behavior. We also propose, in this work, an algorithm based on graph matching concepts and a visualization algorithm able to show the recovery process in time. As a result, we obtain a graph, tracking the facial nerve stream. Then, physicians can observe the recovery progress. Hence, identifing the occurrence of conduction problem preventing it.**

Keywords-Bell’s palsy recovery; Matching Algorithm; Moving Object; Visualization

I. INTRODUCTION

The body's networks allow sub-systems (i.e., digestive, brain, pulmonary) to run independently passing key information when needed. Once this information is blocked somewhere, it implies a dysfunction of the sub systems, which is diseases’ reason. In addition to that, the human body is a complex system succeptible to rapidly change; such instability may also give rise to severe disease.

Among these sub-systems, the facial nerve the failure of which causes facial paralysis that, despite the techniques that are used to accelerate recovery, effective treatment is not yet well defined. The treatment of Bell's palsy is variable [8] ranging from observation to surgical decompression. Among these treatments we opts for electromyography (EMG). The advantages of EMG [9] include that it is relatively inexpensive and is performed by a machine. It yields a lot of data that is continuous and scalar, increasing its apparent credibility. It can detect more subtle muscular activity than visual measurement, and is the only useful approach when movement is not visible. At each patient’s visit, doctors apply EMG until patients’ are completely recovering. This code is comprehensible by specialists and physicians but most of times patient doesn’t understand specialized language concerned with medicines which is characterized by pretentious syntax, vocabulary, meaning or graphics. We aim in this research to simplify the codification of the facial nerve and ensure its comprehension by patients.

To make our solution evolutionary, the Bell’s palsy patient recovery surveillance, we want to help physicians and care givers to ameliorate their treatment methodology; we adopted from a first an algorithm that produces a modeling for the facial nerve: a colored tree indicating segments in which the stream nerve is operational. In a second, for Bell’s palsy recovery process, we adjust the matching theory to graphs generated by facial nerve modeling to detect graphs commonalities and differences: this is ensured thanks to a matching algorithm.

The remainder of this paper is organized as follows: In Section 2, we present a literature review organized in two parts: on the first side, we present facial nerve modeling and on the one second side we discuss some matching algorithms and make a comparison between them. In Section 3, once facial nerve anatomy is well understood, we propose a Facial Nerve Modelisation in the aim to adjust graph matching to these graphs. Then, in Section 4, based on this model, a visualization algorithm was proposed. And finally, in Section 5, we summarize the work and propose new perspectives to be done in the future.

II. RELATED WORK

A. *Bell’s Palsy Modeling*

Understanding the facial nerve anatomy was essential to reach our objective. Akaichi et al. [1] aim at supervising the states evolution of patients affected by facial paralysis leading to recovery. Modeling facial nerve stream trajectory data seems to be an essential step leading to perform our purpose. Moreover, visualizing the facial stream nerve trajectories may help physicians to understand deeply the disease through comparisons performed on patient state in time, or between different patient states. Among facial nerve components, they treated the bronchial motors (the muscles). First, a modeling of muscles using UML [1] class diagram gives rise to a graphic modeling of muscles. This muscles model will serve later to follow the evolution of the bell's palsy patient's state. This is the result of the Facial Nerve (FAN) algorithm [1], which takes as input the graph G representing the facial nerve graph and the muscles responses recoded in a table, and produces a colored graph ColoredG indicating the trajectories of the facial nerve stream. A trajectory in the Bell's palsy case is starting from the End_node linked to the considered muscle until the graph starting node I0. Green trajectories demonstrating that the facial nerve stream crosses this trajectory and a normal function of the muscle. Unlike to a red trajectory indicating a dysfunction of the muscle, hence the cut of the stream somewhere. This algorithm has a spatial and temporal exponential complexity. This is due to the storage of graphs at each medical examination and the pass through the graph when coloring trajectories. In this paper, we adopt the muscles modeling and extent it to add the visceral motor components (Glands) modeling and in a second we minimize the FAN algorithm complexity to a linear one. This will be discussed in details in Section 4.

B. *Matching Algorithm*

Homomorphism [10] is proven helpful in many areas; to apply it between graphs several algorithms have been proposed. Before proposing the algorithm appropriate to the Bell's palsy case, we present in the following some algorithm proposed in the literature. Among the basic algorithms, we quote the Hopcroft-Karp Algorithm originally was invented by John Hopcroft and Richard Karp in 1973 and has ever since been vital for computer science. The Hopcroft–Karp algorithm [2] is an algorithm that takes as input a bipartite graph and produces as output a maximum cardinality matching.

It runs in O ($\sqrt{|V|*|E|}$) time in the worst case with E is the number of edges in the graph, and V is the number of vertices of the graph. Hopcroft-Karp is one of the fastest algorithms that find the maximum cardinality matching on a bipartite graph. The algorithm uses the augmenting path technique as well. However, in order to speed up the working time instead of searching for paths one by one it looks for many paths in the same time. The way the algorithm works is that it continually increases the size of a partial matching by discovering and utilizing augment paths. The main idea is to guarantee that the length of the path grows in each step.

Afterwards, Haloui and Wang [3] propose a new graph matching algorithm for computing the similarity between graphs which proposes a novel approach to the search for the best matching between two graphs. The search process is decomposed into K phases. The promising mappings in each phase are extracted and their matching errors are computed. Given two graphs, the goal is to find the best matching between their nodes that leads to the smallest matching error. This matching error is computed by the dissimilarity between each pair of matching nodes added to the dissimilarity between corresponding edges. An E*V matrix (P) is introduced such E is number of edges and V is the number of vertex. A Pij element in P denotes the dissimilarity between i and j in two graphs.

Recall that graph matching can be applied to several fields. Later on, Remco et al. [4] focus on the application of graph matching algorithms to this similarity search problem. Answering a similarity search query involves determining the degree of similarity between the search model and each model in the repository. Similarity in this case can be defined from several perspectives, including the following:

- Text similarity: based on a comparison of the labels that appear in the process models (task labels, event labels, etc.), using either syntactic or semantic similarity metrics, or a combination of both.
- Structural similarity: based on the topology of the process models seen as graphs, possibly taking into account text similarity as well.
- Behavioral similarity: based on the execution semantics of process models.

The graph matching algorithms studied in this paper attempt to establish 1-to-1 correspondences between nodes in the compared process models.

The problem of measuring object similarity can turns into a problem of measuring object similarity turns into the problem of computing the similarity of graphs, which is also known as graph matching. In this paper, application of graph matching will be demonstrated giving examples from the fields of pattern recognition and computer vision.

The run time of this algorithm is O (βV^{2V+1}), where V is the number of vertexes in the input graph and β is a threshold that defines the maximum number of admissible edit operations. Therefore, this approach is limited to (very) small graphs. The graph matching algorithms reviewed in this paper are very general. In fact, there are no problem dependent assumptions included. The nodes and edges of a graph may represent anything, and there are no restrictions on the node and edge labels.

The graph matching is also applied to the field of image comparison and recognitions. Vertices in graphs represent regions of images, and the division in regions is the result of a segmentation procedure. Hence, automatic segmentation and graph construction techniques are applied to create the graphs that are to be matched. In order to consider a homomorphism as valid, all the vertices in the model graph will have at least a vertex in the data graph that has been matched to it. The complexity of the algorithm depends on the number of phases K. For a given K, to find the best matching we need O (V^2K^V) steps, where V is the number of nodes in the smaller graph.

A comparative study between four algorithm presented above is detailed in Table 1:

TABLE 1. COMPARATIVE STUDY BETWEEN GRAPH MATCHING ALGORITHMS

Algorithm	**Input**	**output**	**Complexity**	**Techniques used**
Hopcroft–Karp	a bipartite graph	a maximum cardinality matching	O(√nm)	The augmenting path
Hlaoui and Wang	A(E*V) matrix	Similarity between graphs	O (V^2K^V)	Similarity matrix
Remco and al.	two business process graphs	Map set between the graphs		Greedy algorithm and pruning
Horst Bunke	A graph and a thresholdβ		O $(\beta V^{2V+1}\ )$	

III. FACIAL NERVE MODELING

A. *Fcial Nerve Components Modeling*

Understanding the facial nerve anatomy [1] is essential to reach the main objective of our work. Indeed, facial nerve can be subdivided into two main components: motor components and sensory components. Essential motor components are the bronchial motor that efferent supplies the muscles of facial expression [6], and visceral motor that vehicles the parasympathetic innervations to all glands of head.

In this paper, we adopt the visceral motor. We subdivide the facial glands into two classes: the superior half glands and the lower half glands. The superior half glands are the eye glands which is the Lachrymal Gland (LG).

The lower half glands are ear glands, nose glands and the salivary glands. The ear glands are called ceremonious gland (CG). Nose glands are Nasal Glands (NG) and salivary glands which are composed by cheek glands and mouth glands. Cheek glands are the Parotid Glands (PG) and mouth glands subdivided into Sub maxillary Glands (SmG) and Sublingual Glands (SlG). Figure 1 describes visceral motor components details using UML modeling.

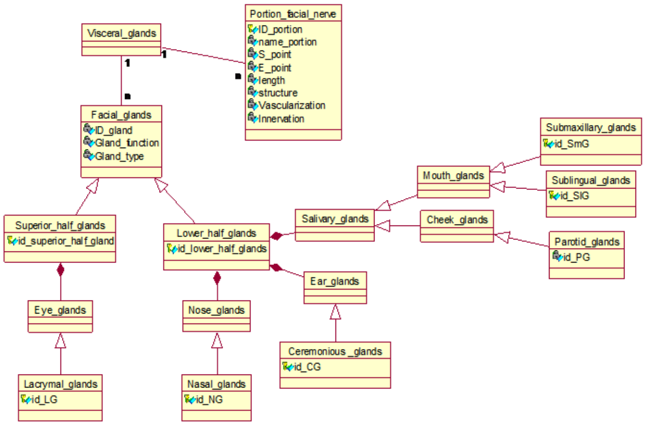


Figure 1. Visceral Motor Component Class Diagram

Understanding facial nerve anatomy permits us to better understand the Bell's palsy disease. Once facial nerve structure is established and facial nerve structure is analyzed. We note that to better understand this disease a graphical modeling of the structure can be the efficient way. A useful way of representing the knowledge is by using graphs. They have been proved as an effective way of representing objects [7]. For the visceral motor components, nodes represent intersection glands and arcs represent the connections between them.

The start node of the graph describes the beginning of the facial nerve of one side (the left side or the right side) of the face, the end-nodes represent facial glands, and arcs describe connections between nodes (Fig. 2).

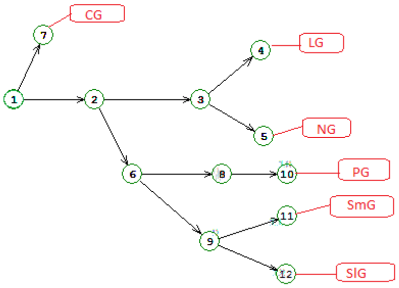


Figure 2. Facial Nerve Glands Graph

To better understand this graph, a mathematical formulation will be more suitable:

A graph G can be defined by a couple G (V, E) in which:

- V(G) is a set of nodes or vertexes:
  - Vertex (the root): the facial nerve beginning with a degree superior to 1 which the degree of a vertex is the number of edges that connect to it.
  - Internal nodes: facial nerve bifurcation with a degree superior to 1.
  - External nodes (or leaf): facial muscles or facial glands with a degree equal to 1 except the WM muscle.
- E(G) is a set of edges such that each edge eij=vi.vj connects nodes vi and vj where is a set of element pairs V. The edges here represent facial nerve portions which is the connection between nodes in wherein facial nerve stream circulates. The facial nerve stream circulates in one direction (a directed graph), this why we have an ordered pair.

The graph G can also be characterized by:

- A path: a unique sequence of nodes and is an alternating sequence of vertices and edges, beginning and ending with a vertex which in our case represent the trajectory of facial nerve stream from the facial nerve beginning until reaching a muscle or a gland.

- The distance (D) of a path from vi to vn is measured by the length of the unique set of edges implicitly defined by the path. The length of a path here is the number of facial nerve portions composing the trajectory. D = V–1, where V is the number of vertexes visited (a vertex is counted each time it is visited). In our case is the number of facial nerve bifurcation.

B. MFGS Algorithm For Bell’s Palsy State Evolution

Recall that the objective of this modeling is to supervise the patients’ state evolution and see the recovery at each medical examination. The recovery is determined after a comparison between muscles and glands intensities computed using EMG and ENoG respectively of both sides: paralyzed and healthy. For two consecutive medical examinations, the evaluation of the disease is due to the comparison process between paralyzed graph obtained at the last medical examination and the current one of the paralyzed side.

Hence, the problem of supervising the patient’s evolution turns into the problem of computing the similarity of graphs between the healthy side and paralyzed side, which is also known as graph matching.

Analogously to Bell’s palsy, physicians need to match the muscle of the paralyzed side to those in healthy side to assess patients’ disease recovery. This matching is based on the field intensity of each node more especially each leaf.

Given two node-labeled graphs G1 = (V1, E1) and G2 = (V2, E2), the problem of graph homomorphism is to find a mapping from V1 to V2 such that each node in V1 is mapped to a node in V2 with the same label, and each edge in E1 is mapped to an edge in E2 from a model graph to a data graph. The sense of matching depends on the healthy side and paralyzed side.

If there is a homomorphism from a graph G1 to a graph G2 we say that G1 maps to G2 and we write simply Φ: G1 → G2 which maps vertices to vertices and edges to edges.

In Bell’s palsy application, matching a vertex to other one means automatically matching the whole trajectory or path from the leaf until the vertex. The matching is determined after a comparison of intensities. If we map a vertex in the model graph to other one on the data graph, this means that the muscle or gland has a normal function and the facial nerve stream crosses the entire path. Hence, all the edges belonging to the muscle path are matched automatically.

To connect two nodes, the following condition must be satisfied: the intensity of the leaf which refers in this case to a facial muscle or gland on the paralyzed side must be greater than or equal to that on the healthy side with an error margin.

The graph homomorphism is used to detect some metrics. Among them, we cite the measure graph similarity.

By analogy to Bell’s palsy, graph homomorphism can be used to detect the disease progression. At each visit, EMG is applied to facial muscle for both sides. Thanks to a comparison of intensity for the same muscle at each side, we conclude the recovery or not for the muscle. Mixing EMG and homomorphism, physicians can detect the Bell’s palsy recovery:

- The total recovery implies a complete matching from the healthy side’s graph to the paralyzed side’s graph
- The partial recovery implies a partial matching of two sides.

The number of nodes matched measure the graphs similarity. This metrics is based on the maximum of nodes matched; a completely matching or a homomorphism implies all the nodes in the paralyzed graph (data graph) are matched to nodes on the healthy graph (model graph) and finally as a conclusion we have a total recovery.

For finding matching with the maximum cardinality or the maximum graph similarity, we propose this algorithm:

```
MFGS Algorithm
  Input: Two labeled graphs for healthy (Gh) and paralyzed (Gp) side
  Output: Matching between nodes in Gh and Gp from the data graph to the model graph For each leaf in the paralyzed graph
1. Begin
    2. HealthyLeaf=BFS(Paralyzed_graph.Leaf, Healthy_graph)
    3. If (Compare (Paralyzed_graph.Leaf. Intensity, Healthy_graph.Leaf.Intensity))// a valid mapping
       Match (ParalyzedLeaf, HealthyLeaf)
       Match(ParalyzedLeaf.trajectory, HealthyLeaf.trajectory)
       Increment C
      4. If (Total matching)
       Total recovery
    5. Else
       Display Sub Graph matched
       Partial recovery
    6. Return C
7. End
```

The BFS function (Breadth First Search) is a strategy for searching in a graph when search is limited to essentially two operations: (a) visit and inspect a node of a graph; (b) gain access to visit the nodes that neighbor the currently visited node [5].

The BFS algorithm is as follows:

```
BFS Algorithm
1. Enqueue the root node
2. Dequeue a node and examine it
    • If the element sought is found in this node, quit
      the search and return a result.
    • Otherwise enqueue any successors (the direct
      child nodes) that have not yet been discovered.
3. If the queue is empty, every node on the graph has
   been examined – quit the search and return "not
   found".
4. If the queue is not empty, repeat from Step 2.
```

Matching algorithms are used to determine commonalities and differences between two structures. Differences can be due to the inequality of intensity of muscle for both sides.

Hence, the evaluation of commonalities gives raise of the structural similarity and the Bell's palsy's recovery. The oriented graph or the tree can be presented as an XML document. Hence, patient's arborescence is saved in the form of a XML document. The above figure represents a part of an XML document for a patient at a visit:

```
<?xml version="1.0" encoding="UTF-8"?>
<Patient>
  <Name>Jabali Mariem</Name>
  <BirthDate>09-09-2005</BirthDate>
  <Gender>Female</Gender>
  <MaritalStatus>Single</MaritalStatus>
  <Location>Borj cedria</Location>
  <ParalyzedSide>Right</ParalyzedSide>
  <ParalyzesCause> dropped from a height of 3 m
</ParalyzesCause>
  <MedicalHistory>allergic to nothing</MedicalHistory>
  <Muscle classe="P2">
    <name>HED</name>
    <Side>Right</Side>
    <type>frontal muscle</type>
    <intensity>45</intensity>
  </Muscle>
</Patient>
```

The matching algorithm is exploited for the detection of the similarity of XML document. Evaluating these similarities is relevant for detecting the patient evolution and the degree of reaching of disease. Then, the similarities measured can be exploited for grouping together patient having the same characteristic. In order to obtain the best match between the two structures, common data contents must be maximal. Whereas, common data contents refer to same intensity if same muscle or glands for both sides: healthy or paralyzed. Then, we want to obtain a numeric value that quantifies the similarity between both sides. The evaluation similarity function is R. This function computes the ratio between the evaluations of common data contents C between the two structures (number of element having same intensity considering the error margin) and the evaluation of all elements A in the two structures (total number of muscles or glands). The obtained similarity value is a real number in the range [0, 1].

$$R=\frac{C}{A}$$

$$\begin{cases} R=0 & \text{if there's no similarity between two structures then minimal matching then totally paralyzed.} \\ R=1 & \text{if there's a perfect similarity between two structures then maximal matching then totally recovered.} \end{cases}$$

A set of node pairs (x, y), M is called a matching from Gp to Gh iff:

- (x, y) ϵ M, x ϵ Gp, y ϵ Gh, signature (x)= signature (y)
- Qqs (x1, y1) ϵ M and (x2, y2) ϵ M; x1=x2 iff y1=y2 then one to one matching
- Qqs (x, y) ϵ M suppose x1 is the parent of x
  Y1 is the parent of y

Then x1, y1) ϵ M → M preserves ancestor relationships

- Suppose (x1, y1) ϵ M } x1 is an ancestor of x2 iff
  (x2, y2) ϵ M } y1 is an ancestor of y2

There are several steps in the algorithm:

1. Determining the signature of each nodes for both XML structures
2. Determining the matching set M
3. Generating the evaluation similarity function

Let us present the three phases in details.

Determining the signature of each node consists on browsing the XML document and for each element (muscle or gland) extracts the value of the attribute name and the value of the intensity, then concatenates them. This function returns a set of signature (string).

```
  Extraction Algorithm
Input: an XML document
Output: a set of signature
1. Begin
    2.    For each line in the XML document
    3.    Extract element labels: name and value
    Concat two lables
    Save the signature// save the signature in S
    Return S
End
```

Once all signatures are obtained, the matching function is applied.

```
MFGSI Algorithm
Input: two labeled graph and two sets of signature S
Output: a set of matching M
Begin
    M1= {all leaf nodes in Gh}
    M2= {all leaf nodes in Gp}
    Do {
    For every node x in M1
    For every node y in M2
        • If (signature (x)= signature (y))
        • Save matching (x,y) // save the matching in M
    Set M1=( parents node for previous nodes in M1}
    Set M2=( parents node for previous nodes in M2}
    }while both MA and M2 are not empty
    Return M
End.
```

In the case of partial matching, which refers to partial recovery, the algorithm may return the homomorphism sub graphs. By definition, given two graphs G and H as input, the sub graph homomorphism which are structurally and text similar. Analogously to Bell's palsy, the sub graphs homomorphism refers to facial part which is not affected by the disease. Later on, these sub graphs can be used for patient classifications.

C. *Example*

In the following, we present an execution example for a patient at each medical examination. So we can see the patient's state evolution through the graphs matching:

At the first visit, doctors detect a Bell's palsy in the left side. Hence, EMG is applied to both sides. As the left is affected, physicians use intensity computed in the right side as a threshold.

Applying the algorithm we obtain this matching, matched nodes implies the matching of the whole trajectory from the root until the leaf (Muscle or glands). In this example we focus on the muscles.

During the first examination, physicians detect with some exercises the incapability of smiling and an asymmetrical smile, light sensitivity, the difficulty of blinking, and difficulties when speaking. These muscles weakness are proved later using EMG. Once EMG is applied, two labeled graphs are created, one for each side containing muscles characteristics which are: name and intensity. The goal is to find the matching set M and evaluating similarity.

The first step is to store the muscle intensities in an XML file:

```
<?xml version="1.0" encoding="UTF-8"?>
<Patient>
<Name>Jabali Mariem</Name>
<BirthDate>09-09-2005</BirthDate>
<Gender>Female</Gender>
<MaritalStatus>Single</MaritalStatus>
<Location>Borj cedria</Location>
<ParalyzedSide>Right</ParalyzedSide>
<ParalyzesCause> dropped from a height of 3 m
</ParalyzesCause>
<MedicalHistory>allergic to nothing</MedicalHistory>
<Muscle Date="January 25, 2007">
<name>HED</name>
<Side>Right</Side>
<type>ChM</type>
<intensity>75</intensity>
<type>EM</type>
<intensity>50</intensity>
<type>EyM</type>
<intensity>35</intensity>
<type>FM</type>
<intensity>30</intensity>
</Muscle>
</Patient>
```

Then, we compare the muscles intensities. To do this, signature of each muscle has to be created on which then the comparison is based.

The sets of signature:

S1={ChM75, EM50, EyM35, FM30, HAM76, LLD98, LLM81, MAD87, MZM95, NeM54, NoM90, NsM91, PoAM110, PrAM135, RM78, TM70, ULM53, SZM76, WM100}

S2={ChM88, EM101, EyM90, FM94, HAM76, LLD120, LLM121, MAD86, MZM93, NeM92, NoM87, NsM88, PoAM105, PrAM132, RM76, TM78, ULM88, SZM101, WM96}

These two sets M1 and M2 are the input of the matching algorithm added to the two labeled graphs. When the algorithm is over, we obtain as output a set of matching:

M={{ChM75,ChM88};{HAM76,HAM76};
{MAD87,MAD86};{MzM95,MzM93};{NoM90,NoM87};

{NsM91,NsM88};{PoAM110,PoAM};{PrAM135,PrAM132};{RM78,RM78};{WM100, WM96}}

Then the ratio of similarity is computed:

$$R = \frac{\text{number of couples nodes matched}}{\text{number of couples nodes}} = \frac{10}{19} = 0.52$$

This ratio determine the gravity of Bell's palsy, in this case the patient have a paralysis with 0.52 gravity.

In addition to the sets of matching, we can display the matching using graphs as shown in Figure 3:

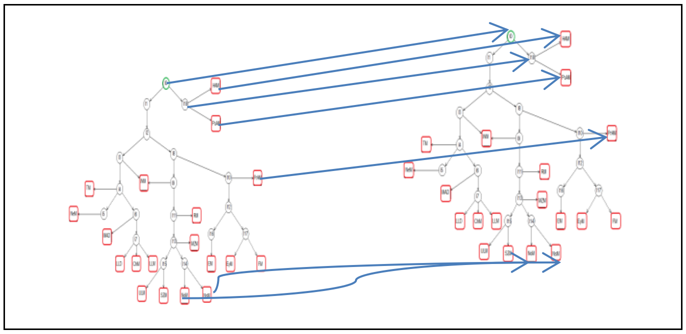


Figure 3. Matching between healthy and paralyzed side

This algorithm is applied at every medical examination. At the second one, we keep the same sets of matching by adding some couples recovered after treatments.

M= M U {{FM94, FM94}; {EyM90, EyM90}; {EM50, EM50}; {ULM89, ULM88}; {SzM101, SzM101}}

$$R = \frac{15}{19} = 0.78$$

Finally, at final medical examination, which corresponds to a totally recovered patients we have all nodes matched and a ratio equal to 1.

Collecting ratios computed at each medical examination, it can show the disease gravity evolution through the time. This is explained by a designed using Matlab in Figure 4:

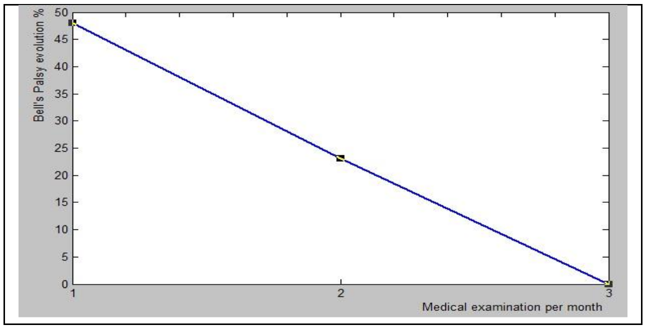


Figure 4. Disease Gravity Evolution

IV. FANI Algorithm vs. FAN Algorithm

For FAN Algorithm [1], for each execution, it has as input a graph G refers to the facial nerve structure and a table containing muscles or glands intensities and their thresholds. As output we display and store a colored graph indicating the recovery process. In terms of spatial and temporal complexity this can costs a lot. The temporal and spatial complexity when dealing with trees depends on the number of nodes. Suppose V nodes' number; the complexity is 2^V, hence this complexity is exponential. We aim in this part to optimize the complexity of the previously proposed algorithm.

The storage of patients' trees is expensive in terms of spatial complexity due to the data structure used and a high time complexity when following the entire route of the trees at each medical examination. Knowing that all patients have the same facial nerve structure so same tree structure, the differences on the trees are localized at the last layer. This layer consists of trees' leaves: muscle or glands and their characteristics. Among the characteristics, the most important one are muscles or glands name and intensities. For coping to the complexity issue and based on structure commonalities and differences, the solution is to store one and only one generic tree without her last layer, which means without leaves which refers to muscles or glands. For muscles or glands, storage is done in forms of table containing muscle name and intensities.

For each patient and at each medical examination, an intensities table is stored containing muscle or glands intensity and the examination date, and then comparison is between two tables of two different consecutive medical examinations. Hence, once the visualization process is triggered, the tree is created by combining two parts: the common part that is saved to the generic structure and the table containing the intensities representing the leaves of this tree. Thanks to this storage, the complexity of intensities complexity has decreased from an exponential to a linear complexity: from O (2^V) to O (V) with v is nodes' number.

Following these changes, the FAN algorithm has evolved. It takes as input two intensities table: T1 and T2; T1 contains the intensities of the paralyzed side and T2 contains the intensities of the healthy side called the threshold. Than the comparison is based on those two tables. Once, the user wants to see the patient's state, the generic graph is concatenated with muscles stored in table 1 and based on the comparison the colored graph is displayed; so the backup is only for tables containing intensities and threshold.

The algorithm is as follows:

```
Algorithm FANI
Input: Table1, Table2
Output: a colored graph
1. Begin
    2. Compare ( Table1, Table2)
    3. Concat ( GenericGraph, Table 1)
    4.  For each leaf in graph
    5.  Color_trajectory( leaf, root, red)
    6.  Color_trajectory( leaf, root, green)
7.  End.
```

For each patient and for each medical examination an intensities table is stored:

```
Algorithm Storage
1.  Begin
    2.  For each medical examination
    3.  Compute intensities
    4.  Save in Table1
    5.  Compute muscle threshold
    6.  Save in Table2
    7.  Return Table1, Table2
8.  End.
```

EXAMPLE:

We took the same case presented above in the matching theory.

The intensities computed are stored under this format: the intensities are ordered in an alphabetical order of muscles:

ChM	EM	EyM	FM	HAM	LLD	LLM	MAD	MZM	NeM

NoM	NsM	PoAM	PrAM	RM	TM	ULM	SZM	WM

Once the EMG technique is applied to all facial muscles; we obtain these values stored in two different tables. Each one for a special sides: paralyzed that will serve for comparison and healthy which will serve as threshold.

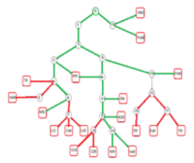
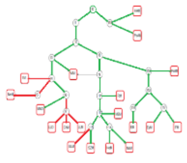
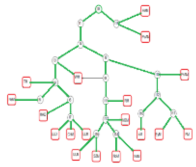
75	50	35	30	76	98	81	87	95	54

90	91	110	135	78	70	53	76	100

88	101	90	94	76	120	121	86	93	92

87	88	105	132	76	78	88	101	96

To supervise the patients' state, the colored graphs can be displayed in order of medical examinations.

		
25 January 2007	25 February 2007	24 March 2007

V. CONCLUSION AND FUTURE WORK

Bell's palsy disease is the origin of a physical suffering, and has an emotional and psychological impact on patients. To contribute in the improvements of treatments and analysis automation, we proposed clear and concise modeling, we also represent it using a graph leading to track facial nerve dream and to determine by the way patients' recovery progress and eventual conduction problems to be solved thanks to observation. This is ensured thanks to an algorithm based on matching theory which produces as output similarities between graphs and the disease gravity.

Future work will focus on integrating manipulated data resulting from treatments performed on various patients by a range of physicians in a various health care institutions. This, obviously, will enhance analysis and large-scale exploitation of these data which is difficult and complex.

REFERENCES

[1] J. Akaichi, H. Bouali, and Z. Dhouioui, "Facial nerve stream trajectory data modelling and visualization," Worldcomp Conference, Las Vegas Nevada USA, pp.101-105, July 2011.

[2] S. KHuller, "Hopcroft–Karp matching algorithm," Lecture 6 in CMSC Advanced Algorithms, February 2002.

[3] A. Haloui and S.Wang, "A new algorithm for graph matching with application to content based on image retrieval," International Workshop on Structural, Syntactic, and Statistical Pattern Recognition, Canada, pp.291-300, 2002.

[4] D. Remco, D. Marlon, and G. Luciano, "Graph matching algorithm for business process model similarity search," BPM '09 Proceedings of the 7th International Conference on Business Process Management, Germany, pp.58-63, 2009.

[5] K. Mehlhorn and U. Meyer, "External-Memory Breadth-First Search with Sublinear I/O," ALGORITHMS — ESA, Lecture Notes in Computer Science, Rome, Italy, vol. 2461/2002, pp.21-26, 2002

[6] M.V. Ramasamy, "Ram's Textbook of Human Anatomy for dental students," Jitendar P Jaypee Brothers Medical Publisher, 2009

[7] M. A. Eshera and K.S. Fu," An Image Understanding System Using Attributed Symbolic Representation and Inexact Graph-Matching," IEEE Computer Science, Poland, vol. 8 no. 5 pp. 604-618, 1986.

[8] D. Kelly, J. Sweeney, T. Jeffrey, and K. Vrabec, "Facial Nerve Paralysis," Department of Otolaryngology, UTMB, Grand Rounds, 1996

[9] J. Hager, "Face and gesture DAQ for Computer Scientists," Lecture for the IWAFGR, France, vol. 905, pp.444,-448, 1995.

[10] G. Hahn and G. Sabudissi, "Graph Symmetry," Lecture Note in Klewer Academic Publishers, pp.107-166, 1997.