École Polytechnique de l'Université de Tours
64, Avenue Jean Portalis
37200 TOURS, FRANCE
Tél. +33 (0)2 47 36 14 14
www.polytech.univ-tours.fr

Computer Aided Decision Support
International Research Master 2
2013 - 2014

Pattern recognition project draft

Multi-scale graph comparison

Supervisors
Romain RAVEREAUX
romain.raveaux@univ-tours.fr

Student
Thomas NOGUER
thomas.noguer@etu.univ-tours.fr

Université François-Rabelais, Tours

M2RI CADS 2013 - 2014

Version of January 18, 2014

Contents

List of Figures

List of Tables

Introduction

When we wish to do the comparison of two graphs, we can come to a limitation when handling high sized graphs. The community detection algorithms allow to simplify a graph by finding communities of highly related nodes. It is then legitimate to want to use such algorithm to reduce the size of graphs in order to perform calculations that are greedy for computation time.

For this project we choose to use a community detection algorithm for graph comparison. The Louvain's method is very fast and easy to implement. The graph edit distance method is known to be very slow when dealing with high scaled graphs. In this project we want to use the community detection to simplify the graphs to be used in the graph edit distance so we can still compare them with a good computation time.

There is still an issue to solve, how do we use one method with the other. The Louvain's method can be used in several iterations to make the graph more and more simple. We must find a relation between the edit distance of two graphs at different scales and the edit distance between the unchanged graphs. At which scale do we decide that the edit distance is close enough to our original graph? Can we use the different scales together in order to be closer to the distance between the original graphs? These questions are at the center of our problem.

The rest of the report features the explanation of the graph edit distance and Louvain's method, the elaboration of the multi-scale graph comparison, the results of the project, the perspectives for the future of the project and finally the assessment of this project in regard of my scholarship.

Graph Edit Distance

One way to compare two graphs is to use their edit distance. The amount of addition/suppression we must perform to transform a graph G1 into a graph G2. If G1 and G2 are equal their edit distance will be 0, if G1 and G2 are infinitely different from each other their edit distance will be close to infinity.

This method can be very greedy for computation time if the graphs are big, we usually use approach of this value for such graphs. This is what we will try to perform here, find a evaluation of the edit distance that is easier to find.

We use the following algorithm in order to find the edit distance between two graphs $G1$ and $G2$:

Algorithm 1 Graph Edit Distance

repeat
  for each node $n \in G1$ **do**
    Add every possible transformation n into the search tree.
    Select a path using A^* and heuristics.
  end for
  for each remaining node n of $G2$ **do**
    Add every needed insertion of existing node into the search tree.
    Select a path using A^* and heuristics.
  end for
until The path is complete
return The cost of the found path.

Community detection: Louvain's method

The community detection is used to find communities of highly correlated nodes within a graph. This method is based on two formulas: the modularity Q and the composite modularity gain ΔQ. The modularity of a graph is found following this formula:

$$Q = \frac{1}{2m}\sum_{i,j}\left[A_{ij} - \frac{k_i k_j}{2m}\right]\delta(c_i, c_j),$$

Where m is the sum of the weights of all the links in the graph ($m = \frac{1}{2}\sum_{ij} A_{ij}$), A_{ij} the weight of the edge between i and j, k_i is the sum of the weights of the edges attached to vertex i, c_i is the community to which vertex i is assigned and finally δ is a function where $\delta(u, v)$ is 1 if $u = v$ and 0 otherwise.

The gain in modularity from moving a node i into a community C is defined by the following formula:

$$\Delta Q = \left[\frac{\sum_{in} + k_{i,in}}{2m} - \left(\frac{\sum_{tot} + k_i}{2m}\right)^2\right] - \left[\frac{\sum_{in}}{2m} - \left(\frac{\sum_{tot}}{2m}\right)^2 - \left(\frac{k_i}{2m}\right)^2\right]$$

Where $\sum_{in}$ is the sum of the weights of the links inside the community C, $\sum_{tot}$ is the sum of the weights of the links incident to nodes in C and $k_{i,in}$ is the sum of the weights of the links from i to nodes in C.

We use the following algorithm in order to find the communities of our graphs:

Algorithm 2 Louvain

```
Assign each node to a unique community.
Compute the initial modularity.
repeat
    for i ∈ V do
        for j ∈ V do
            Remove i from its community and place it into j's.
            Compute the composite modularity gain ΔQ.
        end for
        if There exists a positive gain then
            Choose j with the maximum gain and truly move i to j's community.
        else
            i stays in its community.
        end if
    end for
until No further improvement in modularity
```

Our interest in this method here is that it allows us to subdivide our graph into smaller parts that are highly correlated. It will allow us to transform our graphs to make them simpler.

Multi-scale graph comparision

The main questions remain, how do we use those methods together in order to have a good comparison of graphs? Several approach are possible, do the edit distance at each scale of our graphs and make an average of all the distances. The problem with the majority of the methods we can think about is that they depend on an arbitrary value. What we would like to have is a method that does not depend on a parameter. The method that allow us to do this is to use the classic graph edit distance algorithm on the most simplified graphs by Louvain's method, only that the cost of addition/suppression of a community is equal to its graph edit distance. We have a recursive model that does not depend on the value of a parameter.

We want to compare two graphs G1 and G2. We first use Louvain's method on both graphs until their modularity is not improving, we store the communities of each node at any scale. We then apply the GED algorithm between the most simplified version of both graphs, only when we will evaluate the cost of addition/deletion/substitution of a community (i.e. a node that isn't a node in the original graph G1) we will use the GED of the sub-graph of this community.

In other words, the algorithm will subdivide the calculation of the GED between smaller calculation of GED of parts of our graphs. We hope that this approach will reduce the computation time of the total distance without damaging too much the value of the distance.

Algorithm 3 Multi-scale graph edit distance

```
// The cost function takes the original graphs and the cost of addition/suppresion for a node and an edge.
Cost ← CommunityCostFunction(G1, G2, 1, 1)
// G1 is transformed into its simpler version by Louvain's method.
Louvain(G1)
// G2 is transformed into its simpler version by Louvain's method.
Louvain(G1)
// We call the graph edit distance with the simplified version of our graphs and with the special cost function.
return GraphEditDistance(G1,G2, Cost)
```

The community cost function is the following: let us consider two nodes s and e, we want to get the cost of addition/suppression of these two nodes only that a node can be a community. We use this cost $GED(GetSubGraph(s), GetSubGraph(e))$ where the function $GetSubGraph(c)$ returns the sub-graph on the community c at the previous scale, or c if c is a node.

Results

In order to see the efficiency of our method we chose to compare it with the classic graph edit distance on a set of small graphs (between 5 to 10 nodes). We performed 9560 comparison of different graphs with both GED and multi-scale GED.

GED		Multi-Scale GED	
Cost	Time(ms)	Cost	Time(ms)
1.10	324.63	2.17	8.85

Table 5.1: Average results for GED and Multi-Scale GED

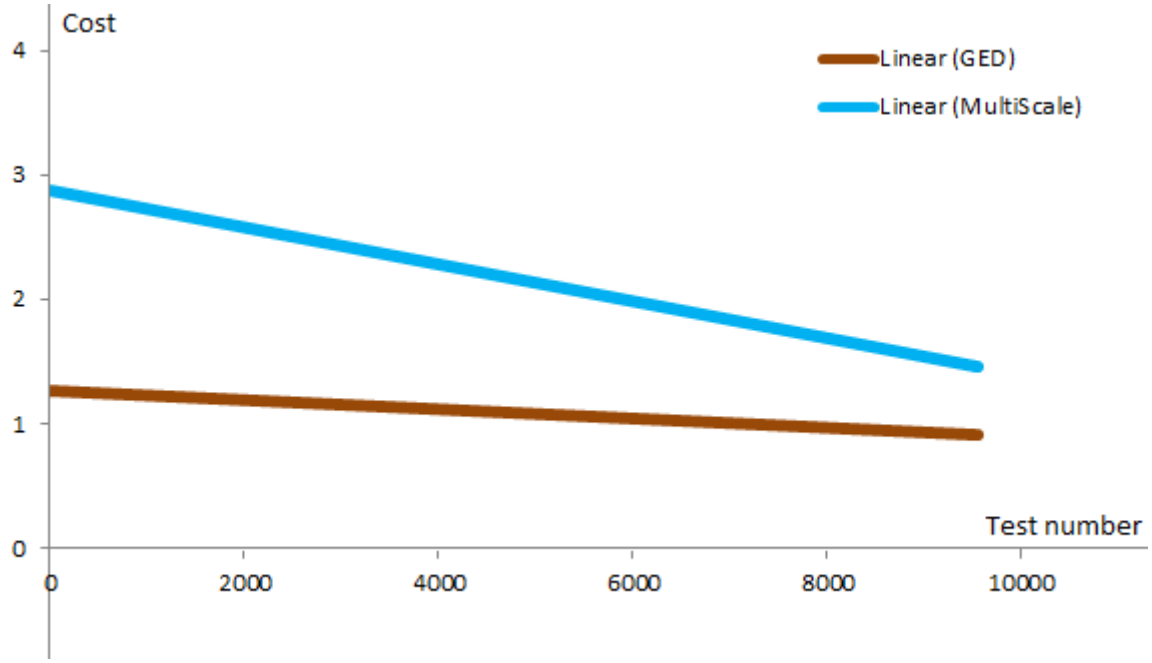

Figure 5.1: The linear functions of the costs.

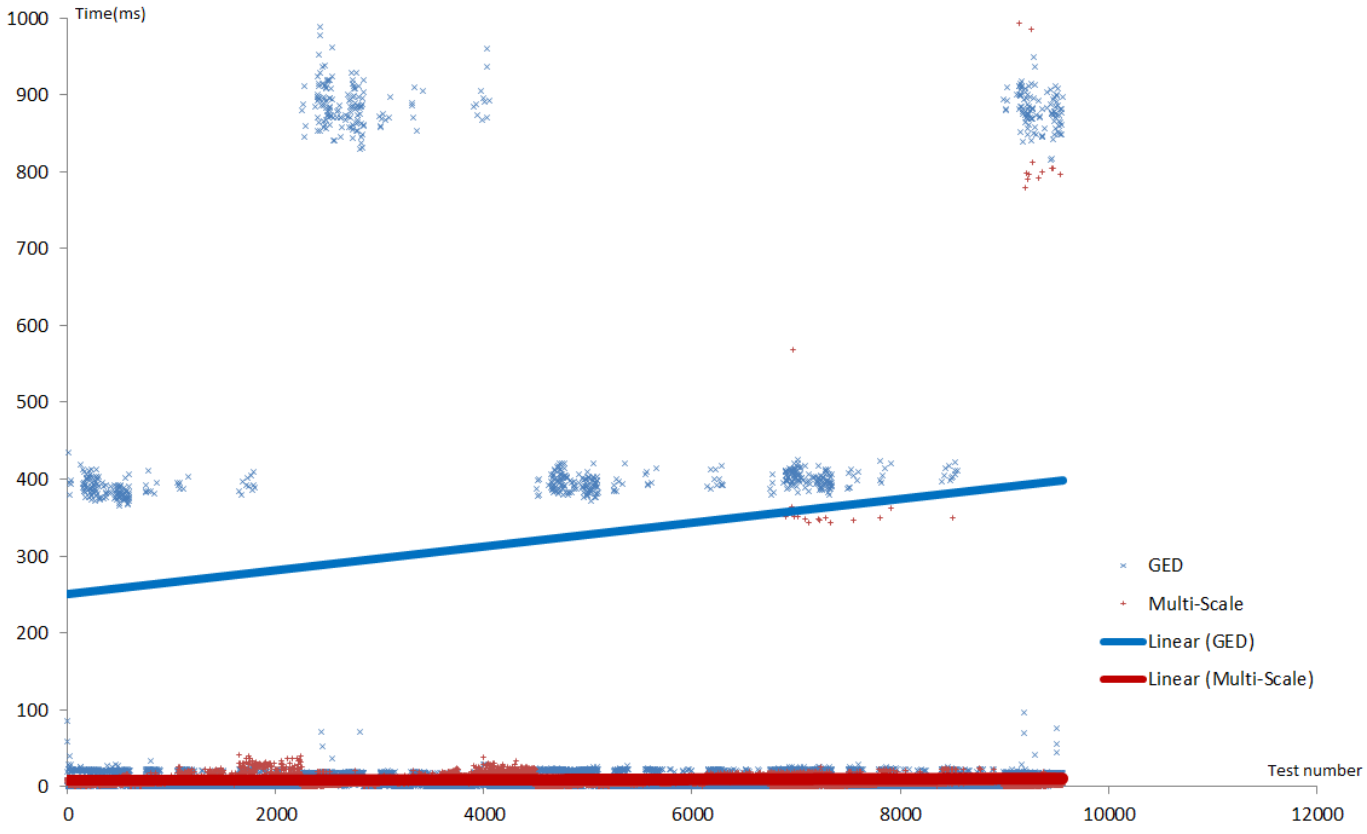

Figure 5.2: The linear functions of the times.

Perspectives

Assessment

Conclusion

Multi-scale graph comparison

Computer Aided Decision Support
International Research Master 2
2013 - 2014

Pattern recognition project draft

Abstract: abstract

Keywords: keywords

Supervisors
Romain RAVEREAUX
romain.raveaux@univ-tours.fr

Student
Thomas NOGUER
thomas.noguer@etu.univ-tours.fr

Université François-Rabelais, Tours