Aux portes de l'est

Raymond Guittonneau

Université de Nantes
44000 Nantes
France
raymond.guittonneau@etu.univ-nantes.fr

Abstract. This document is the report of my internship in the compagny NaviExpert. In this document I will, after a short presentation of the company and its country, present the project : Find the differents time of the traffic condition in Poland. Then I will expose the problem and some other informations. In a second time, I will explain my program. How it works and witch choice I did. In the third time, I will show my results and comment them before doing a last analyse.

1 The Company

1.1 the localisation

The company is based at Poznań. Poznań is a town in west Poland. Poznan is the main city of the Województwo Wielkopolskie (Greater Poland Voivodeship in english). I put a map of the Poland on the figure 1. Poznań is the fifth town of Poland and has a population of more than 566 000 people. I also put the emblem of Poznań in the figure 2.

1.2 The compagny

The company name is NaviExpert (see fig 3 for the logo of the comapny) and has been created in 2005. The company offer a navigation system service on mobile phone. The company is still younth but their system is fully operational and the company has allready won some prices in Poland and has been selected in internationnal competitions.

2 Context of the project

When a custumer of the company use the GPS system, the company records some datas on the custumer. With those datas, The company wanted to have a set of time period, in wich the traffics conditions are similars. The compagny gave me a table with all passages of its polish custumers during the year 2008, exept july and august due to the particular condition of thoses month.

Fig. 1. a map of Poland

Fig. 2. The logo of Poznań

Fig. 3. The logo of NaviExpert

2.1 The road graph

All roads of Poland, wich have been represented in the system, are represented by any small route segments wich have an unique id. Each road id has also a list of previous route segment and a list of next route segment

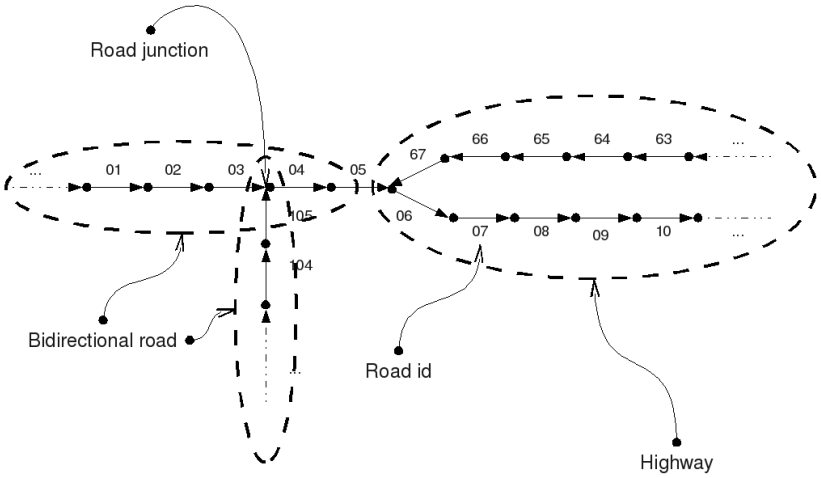

Fig. 4. The graph of the polish roads

The figure 4 represent a part of the polish roads graph used by the NaviExpert's system. It represent (at left) two bidirectionals roads whitch join them before an higthway road (at right). That figure show that the roads are cut in *route segment*. the arrows show that a direction is define for each road. We note that the highways are represent by two chains of *route segment* with an opposite direction ; the highways are represent by two unidirectional roads. If a road is unidirectional, the direction of the route segment is, of course, the direction allowed and no more information is needed on the road graph (for my problem).

2.2 The custumers datas

When a custumer use the NaviExpert GPS system, the compagny records his position on the road graph (the route segment id), his direction (a boolean, since the roads are generally bi-directional), his estamated velocity, the time of his passage and the custumer id.

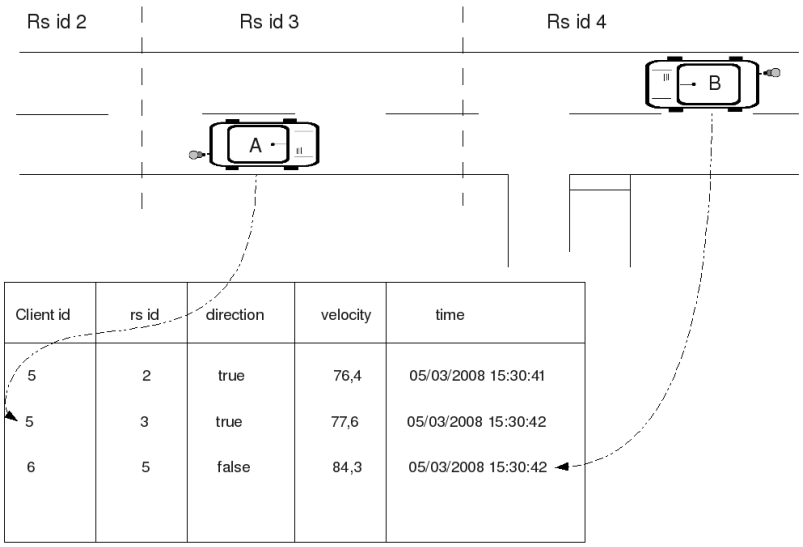

Fig. 5. Passages record

The figure 5 is a draw which show how the compagny records the custumer passages. This exemple show two custumers of the socity, named "A" and "B", driving on a road. This draw shows also a part of the data base. The table of the data base shows that the car "A" is actually on the route segment number 3 and the car "B" is on the route segment 4. On this exemple, the socity records the passage of the two cars. the car "A" has the client id number 5 and the car "B" the number 6. At the moment of the draw (the 5^{th} march 2008 at 15:30:42), the car "A" is actually on the route segment 3 its velocity is 77,6 km/h and its direction is mark as "true". This direction means that the car "A" is driving in the same direction as indicate on the road graph, whereas the car "B" is driving on the opposite side, on the route segment 4, and its direction is mark as "false". The table of passage shows also that the car "A" was on the route segment 2 one second ago, and was driving at 76,4 km/h.

2.3 The problem

To give some better estimation of the time of the travel to their custumer, the compagny wanted to know the time period of the differents traffic conditions, specially the rush periods. This question has appear on a natural observation which is that at the morning and at the evening, the driving conditions are worth than the rest of the day.

To reach this goal, the company gave me the table of all passages of its custumers during the year 2008, except july and august, because those months have naturally some differents traffics conditions of the rest of the year which are not the subject of my work. I have no informations on the position of the road segments, and I can't differentiate the town-roads to the contry-roads. Those informations were not really usefull since it is the variation of the traffic condition that is interesting, and that an arbitral distinction between the town and contry rods could be not corrolated to the traffics conditions. indeed, some contry roads could be influenced by the rush time, and some town roads not.

3 Contribution

The company creates its program principally with the java langage, that's why I used this langage. To differenciate the time periods, we used the velocity information. So we needed to define a distance function between two velocity. This is with this distance that we can say if the traffic condition of a time t_1 is similar or not to a traffic condition of a time t_2. This function will be used to create a bi-dimensional table of distance. This table will be the input of a second algorithm, witch will generate a set of time_period.

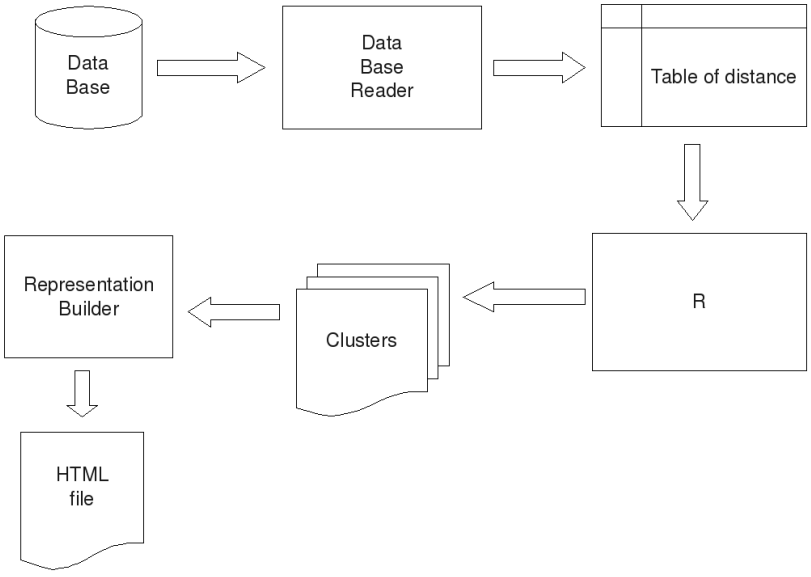

Fig. 6. The application

The figure 6 show the global organisation of the program. The first step is to read the data base, done by the *data base reader* bloc. it produce a *table of distance* which will be the input of the *R analyser* bloc which classify the datas in *clusters*. The last bloc, *Representation Builder* is only use to produce an HTML file of the clusters which is more readable for a human person.

3.1 The table of distance

Due to the large scale of velocities in a simple Time Unit, we choosed to create a bi-dimentionnal table of distance between the velocities of all Time Unit. To do that, we need to define a distance function and a method to compute an average of thoses distances between two Time Unit.

The distance function

The most natural distance is to compute the diference between the velocities. I called this distance *simple distance* and the formula with two velocities v_1 and v_2 is :

$$|v_1 - v_2|$$

This function is very natural, but it give a bigger influence to the contry-roads than the town-roads indeed a difference of 10 km/h in the contry has not the same signification in a town (on a contry road, a velocity of 85km/h is quite similar to 95km/h, despite on a town road, 35km/h is not similar to 45km/h).

To resolve this problem, the second idea is to define the distance function by a ratio between the two velocities. To keep a result between 0 and 1, I define this function, that I called *ratio distance* by this formula :

$$\frac{min(v_1, v_2)}{max(v_1, v_2)}$$

With this function, a difference of 10 km/h is now more important in a town road than in a contry road.(Indeed $ratio\,distance(85, 95) = 0,89$ and $ratio\,distance(35, 45) = 0,77$. With this function, 0 indicate that the two velocities are very differents, and 1 indicate that the velocities are similar.

During this part of my work, my supervisor Andrzej Jaszkiewicz proposed me an other function, that I called *Andrzej distance*, which is the ratio of the difference between the two velocities and the average. The formula is :

$$\frac{|v_1 - v_2|}{(v_1 + v_2)/2}$$

This function produce some results quite similar to the *ratio distance* but the ouput is a number between 0 and 2 , 0 indicate that the velocities are similar and 2 indicate that they are differents.

There is not a lot of difference between the *Andrzej distance* and the *ratio distance*. But the *Andrzej distance* is more sensive on the small differences of velocities than the *ratio distance*. That's why I used the *Andrzej distance*.

The data base reader

With a distance function, we have now to think about how to construct the table of distance. The goal of this table is to show the difference of the traffic conditions between two time unit. The main information able is the velocity. We have now also a distance function to compare two distances. The problem is now how to define a distance between the traffic condition of a time t_1 and the traffic of a time t_2. The most important is to not mix the velocities of a contry road with those of a town road. To resolve this problem, I choose a simple method, I simply compute the average distance of velocities of each couple of time seperatly for each roads.

```
Variables :
velocities[nb time unit]  : an array of list. this array will contain
                                 all the velocities record on one
                                 road store in the right case.

table[nb time unit][nb time unit]   : the table result

begin
   for each road
      fill the array velocity
      for each time t1 :
         for each time t2:
            for each velocities v1 of velocities[t1] :
               for each velocities v2 of velocities[t2] :
                  add in table[t1][t2] the distance d(v1, v2)
end
```

Fig. 7. The algorithm of the construction of the table of distance

On the figure 8, I show how I compute the datas of the table of distance. On the figure 8, I show a part of the table velocities from 8:00 to 8:30. the velocities of this table of list are the velocities of a unique route segment. To fill the table of distance, I simply compute all distance of all velocities. Note that in my algorithm, I use two tables, one for the sum of the distances and a second to count how many distance I did in each cell. With this construction, each couple of velocities has the same weight in the final result.

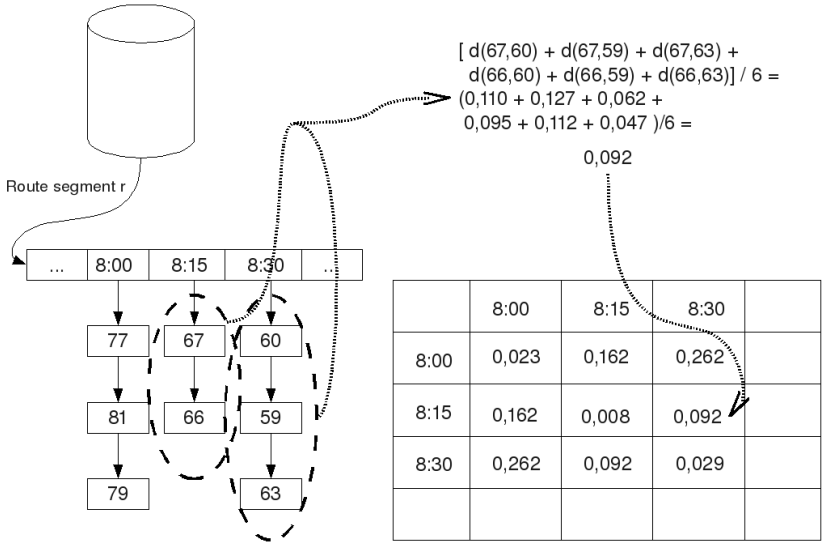

Fig. 8. The construction of the table of distance

3.2 The clustering

Now that we have a table of distance, we have to cluster it. To do that, I use the R program. This program is callable with a simple command line, so it is very easy to call it with a classic programming langage like java.

On the figure 9, I show how I used R. I created an R script creater whitch create a script for R and a text file which contain a copy of the datas of the table of distance. On the figure 10 I show an exemple of an R script. On this exemple, I ask to R seven clusters with the method "hclust ward". Once thoses clusters computed, I used a procedure whitch read the cluster text files and send the results to an HTML builder which create an HTML file with the results expressed in a beautiful format.

The clustering methods provide by R

R provides several methods to clusters some datas. Someone are avalaible natively and some other avalaible via some libraries. I clustered the datas with the native methods and the methods of the library *cluster*.

The native method is *hclust*. The methods of the library that I used are : *agnes*, *diana* and *clara*.

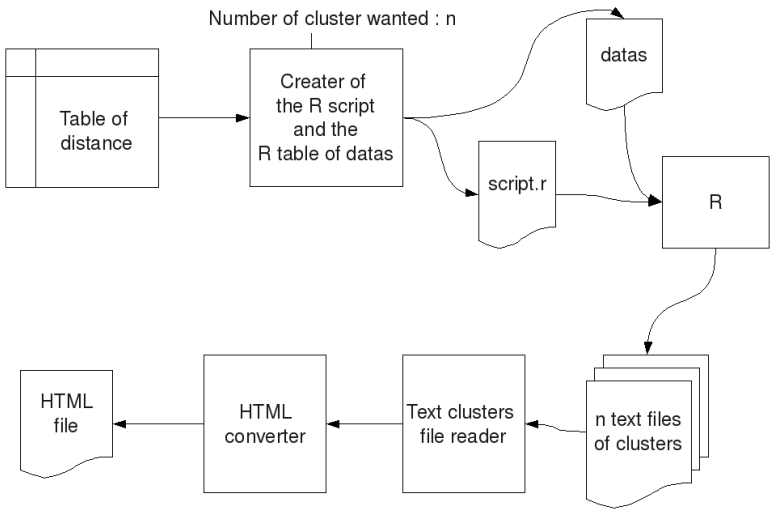

Fig. 9. The application

```
table <- read.table("C:\\Users\\Andrzej\\Desktop\\Raymond\\temp3\\hclust_ward.txt")
table_dis <- dist(table)
hc <- hclust(table_dis, method = "ward")
clust <- cutree(hc, k = 7)
 write(names(clust[clust == 1]),
    "C:\\Users\\Andrzej\\Desktop\\Raymond\\temp3\\cluster_hclust_ward_1.txt")
 write(names(clust[clust == 2]),
    "C:\\Users\\Andrzej\\Desktop\\Raymond\\temp3\\cluster_hclust_ward_2.txt")
 write(names(clust[clust == 3]),
    "C:\\Users\\Andrzej\\Desktop\\Raymond\\temp3\\cluster_hclust_ward_3.txt")
 write(names(clust[clust == 4]),
    "C:\\Users\\Andrzej\\Desktop\\Raymond\\temp3\\cluster_hclust_ward_4.txt")
 write(names(clust[clust == 5]),
    "C:\\Users\\Andrzej\\Desktop\\Raymond\\temp3\\cluster_hclust_ward_5.txt")
 write(names(clust[clust == 6]),
    "C:\\Users\\Andrzej\\Desktop\\Raymond\\temp3\\cluster_hclust_ward_6.txt")
 write(names(clust[clust == 7]),
    "C:\\Users\\Andrzej\\Desktop\\Raymond\\temp3\\cluster_hclust_ward_7.txt")
```

Fig. 10. an R script

The huclust function performs a hierarchical cluster analysis using a set of dissimilarities for the n objects being clustered. The rest of this paragraph is an extract of the R manual page. Initially, each object is assigned to its own cluster and then the algorithm proceeds iteratively, at each stage joining the two most similar clusters, continuing until there is just a single cluster. At each stage distances between clusters are recomputed by the LanceWilliams dissimilarity update formula according to the particular clustering method being used. A number of different clustering methods are provided. $Ward$s minimum variance method aims at finding compact, spherical clusters. The complete linkage method finds similar clusters. The *single* hclust linkage method (which is closely related to the minimal spanning tree) adopts a friends of friends clustering strategy. The other methods(*centroid*, *complete*, *median* and *average*) can be regarded as aiming for clusters with characteristics somewhere between the single and complete link methods.

The agnes function (AGglomerative NESting) is provided by the *cluster* library. This function is also a hierarchical cluster analysis, but its implementation is not the same and so it provide sometime some results a little differents. Due to thoses littles differences, I keep use this function.

The diana function (DIvisive ANAlysis clustering), provided by the *cluster* library, constructs a hierarchy of clusterings, starting with one large cluster containing all n observations. The rest of this paragraph is an extract of the manual page of the library *cluster*. Clusters are divided until each cluster contains only a single observation. At each stage, the cluster with the largest diameter is selected. (The diameter of a cluster is the largest dissimilarity between any two of its observations.) To divide the selected cluster, the algorithm first looks for its most disparate observation (i.e., which has the largest average dissimilarity to the other observations of the selected cluster). This observation initiates the "splinter group". In subsequent steps, the algorithm reassigns observations that are closer to the "splinter group" than to the "old party". The result is a division of the selected cluster into two new clusters.

I used this method to compare the results with the hierarchical clustering methods.

The clara function (Clustering LArge Application), provided by the *cluster* library, works like the $k - means$ algorithm, but is design to work with a lot of data. That's why I selected this function. The $k - means$ algorithm consist to select randomly k objects as center of a cluster and to put all the other object in the cluster in which they are the nearest of the center. then we compute the center of each cluster by using their member and we re-iter n times (except that the centers are now not an object of the datas)

4 Results

With this application, I compute several analyses. I study the traffic condition on a time period of one week with a unit time of 15 minutes. I analyse the traffic condition for one month, one year and one trimester.

4.1 The datas

for my study, I had the table of all custumers passages of the 2008 year execpt july and august. Since the compagny is still younth, I have less datas for junuary 2008 than for december 2008. the number of passage of each month are indicate in the following table.

Table 1. Number of datas available for each month

month	number of passages
january	2 125 820
february	2 337 681
march	2 470 338
april	2 843 053
may	3 122 089
june	4 147 547
september	2 838 879
october	5 525 680
november	5 263 790
december	5 207 619

4.2 The results month by month

In this section I will expose my results month per month. We will observe here that the results of the months from junuary to april don't give us lot of information whereas the results of the next months are more interesting. This is certainly because we have less informations for the first months than for the last. You will also observ a cluster for the times around 3AM or 4AM. This cluster is not a true one, it correspond to the time when the company reboot its server, so those datas should not be interpreted.

The results with 6 clusters are shown on the figures 11, 12 and 13, generally computed with the *hclust* method. On those figure, we don't see the hour, so you have to know that each lines represent a day of the week (from sunday to satursday) and each column represent a period of fifteen minutes from 0:00 to 23:45. Each cells are colored in gray. The clusters and their coloration are sort by the average velocity of the cluster, so the fastest cluster is numbered 0 and colored in black, and the slowest is colored in white and in this case numbered 6.

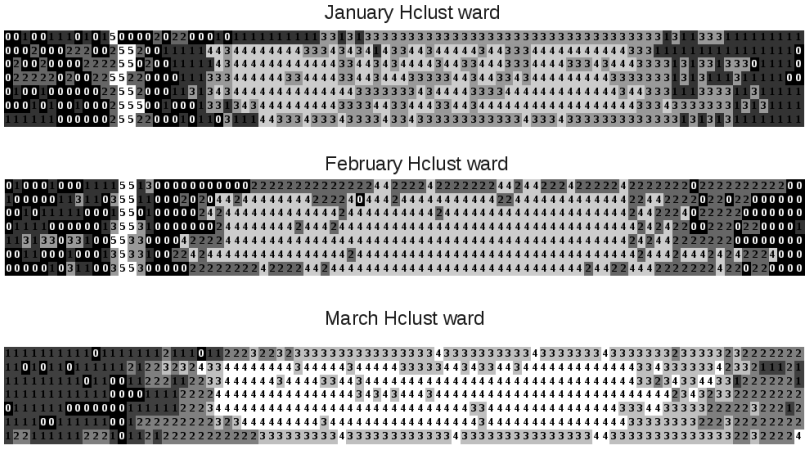

Fig. 11. The results for the months january, february and march

january

Whereas we can't analyse a lot thoses results, we can observ that there is at least two differents traffic condition periods : the day and the night. We can also guess a two other groups : the morning and evening.

february

We can't give a lot of results for this month too but we can clearly look two traffic condition / the night and the day. with thoses results, it is very difficult to see some other clusters.

march

With march, we observ a little difference. we have still two clear cluster but there are a little difference. The first cluster is the nights and week-ends. The second is the days. When we ask to R to classify the datas with more groups we can guess some other clusters like for junuary, but (like junuary) it is not very clear.

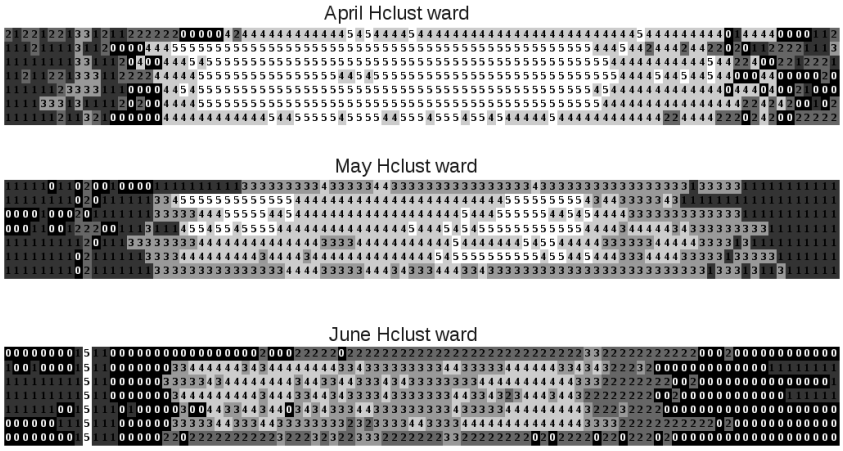

Fig. 12. The results for the months april, may and june

april

In april, we still have two principal cluster, the nights and the week-ends for the first one and the days for the second. But the second can be cut in two more easyly.

may

May is not a month like the other with a lot of unwork days, especially the thursday (and generaly the friday next). Naturaly we observ that in the results. exept for thursday and friday, we have some similar results. The traffic condition of the thursday and friday morning are good like the week-ends. We note that for monday, tuesday and wednesday, the morning and evening rushes appear now clearly

june

With june, we remark that the results begin to be more readable, certainly due to the fact that we begin to have more datas. we have now three cluster that can be read easyly. the night and week-ends, the morning and evening rushes and the noon.

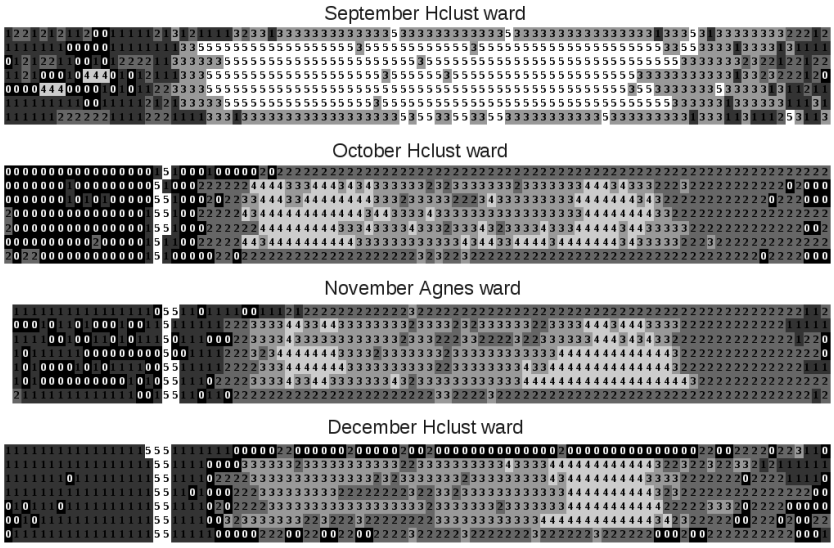

Fig. 13. The results for the months september, october, november and december

september

For september, we don't have a lot of datas, so as for junuary, february and march, we only see the the cluster of the nights and week-ends and the days cluster.

october

With the october's results, we observ clearly three clusters. they are still the same clusters but they appears clearly.

november

For november we have also the same three clusters. We note that on the figure 13, I show the result with the method *agnes*, because this method shows a cluster for the morning rush because the *hclust* method didn't shows it with 6 clusters but it appears when we ask a little more clusters.

december

We also have three cluster. But we the morning rush doesn't appear clearly. When we ask a little more clusters, this cluster appears clearly.

4.3 the results for all the year

After studying the datas month per month, I studied the datas for all the year. Since I have a huge data, I expect that some cluster will appear very clearly. As expected, the clusters have appear very clearly. I show the results in the figure 14. On that figure I show the results for thre differents methods : Hclust ward, Agnes ward and Clara. The morning rush begins around 5:45 and ends around 9:30. The evening rush begin is variable but the variation is constant. The monday it begins around 15:15 and slowly but certainly, it begin earlier untill friday where it begins around 13:30. But the evening rush ends every day around 18:00.

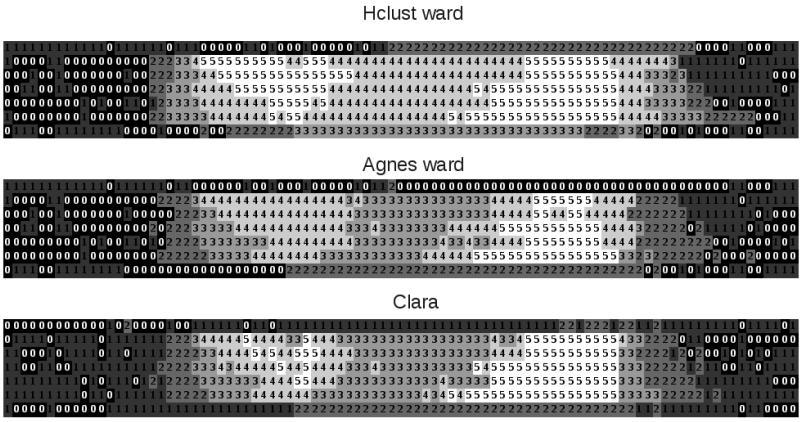

Fig. 14. The results for the all year 2008 with three differents methods.

4.4 The results by trimester

We have seen that the more datas we have the better are the results. but by observing the results month by month, we can see that the time of the rush are not exactly the same in junuary and in june, certainly due to the differents season and the differents length of days. So I compute three other table of results. The first on correspond to the month from junuary to march, the second from april

to june and the last from september to december. With more datas, I expect to obtain some better results for some specifics period of the year. The number of datas for the three periods are indicate by the following table :

Table 2. Number of datas available for each period

period	number of passages
january to march	6 933 839
april to june	10 112 689
september to december	18 835 968

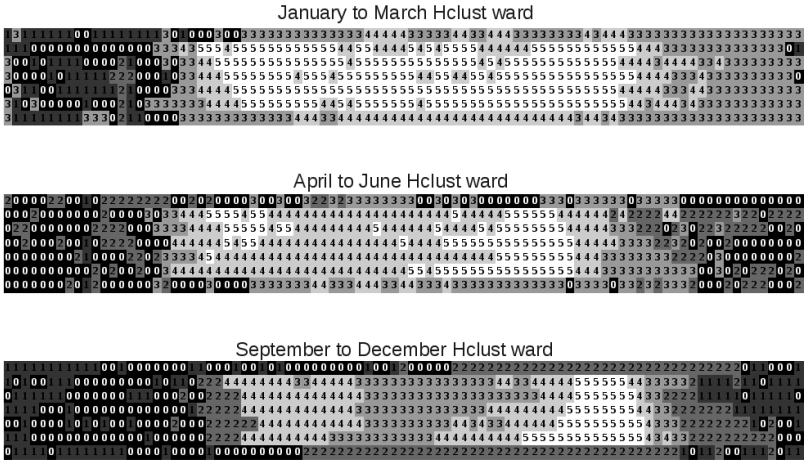

Fig. 15. The results for the months january, february and march

Naturally, I still have less datas for the begining of the year, but I hope that it will be enough since I have more than five million datas which was the number of datas availaible for the last month which were quite good.

The winter, junuary to march

The results for this period are not so good as expected. The results of each month were very differents during the days. So may be that this difference unable the possibility to obtain some good results.

The spring, april to june

The results of the months april, may and june were more similar than the three first, so I still expect some good results. An important thing that will be interesting to observ, is the influence of may in the results. The cluster of the morning and evening rushes appears clearly, but the influence of may, and its unworked thursday, is important, especially for the morning rush.

The autumn, september to december

For this period, I really expect some good results since the results were quite good for each month of this period (exept september). This time, the results are good. We can clearly see the clusters of the morning and evening rushes. We can see that the length of those rush is shorter than thoses viewable with the all year table.We also note that the time of the rushes in the winter are later than the time of spring.

5 Analyse

Once the results compiled, I tried to analyse them. The first interesting thing is the shape of the results.

We can see on this simple view (fig. 16, that the morning rush begins every day a little later, but ends every day at the same time. With the same way, we can see that the evening rush begins every day a little earlier, but still ends at the same time. My interpretation is that people are going to work soon and work also late, at the begin of the week, and day after day, they are going to work a little later and come back to home a little earlier every day, especially at friday. It seem to be realistic, since a large part of people don't go to work for fun, and at friday, they're generally thinking more about their week-end than their job.

I also look that the time of the rush are not the same during the year. The rushes appears sooner at the spring than at the autumn. Despite I have few data at winter and summer, I think that I can suppose without a lot of risk that the fact that the sun appears sooner in summer than in winter influences the time of work of some polish people.

I also tried to look the weather in poland during the year 2008 to look if the weather influance the traffic condition or not. To that, I look the weather in seven towns in all Poland : Szczecin, Gdansk, Suwalski, Warszawa, Kraków, Wroclaw and Poznań (the names are given in polish). I compiled the results month per month to compare it with my results per month. I was limited in my

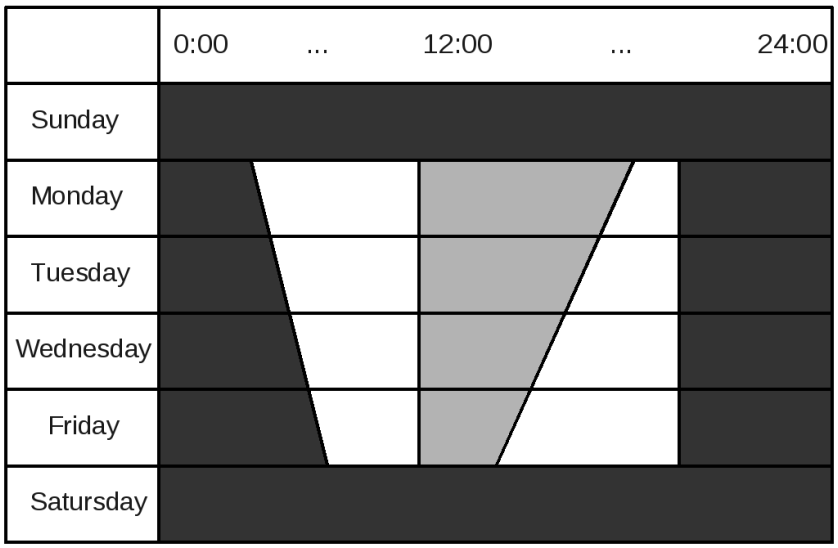

Fig. 16. The shape of the results. On this picture, there is three clusters. the more dark they are, the faster is the average of speed.

analyse by the fact that I had only the weather day per day, i.e. I didn't have the time of when the rain falls. So I can only try to look if one day of the week with a lot of rain has a difference with an other without lot of rain. Unfortunally, I observ not correlations between the weather and the results. In an other hand, this uncorrelation is quite natural because my algorithms are based on a distance between two speeds, and not directly on the velocities.

6 Conclusion

With my analyses, we can't see a good partition of the time, but I used the datas of all the Poland and the rush time of Warszawa (Varsaw), Krakow or Poznan are not necessary the same.So may be some better results could be found with a small part of Poland, like Poznan or Warszawa region. My results are still interessting since I have now a better idea on how people drive, and I know that the length of the rushes has some difference in function of the day of the week.