Internship report:
LIS
Minimal Manhattan network

ABDEL WAHAB Ismail

25 April 2020

Summary

Part I

LIS and intership's subject

Chapter 1

Presentation of the Laboratory

1.1 Creation of the structure

The Computer Science and Systems Laboratory (LIS) in french "Laboratoire d'Informatique et Systèmes", is a structure that is the resulting of the merge of two laboratories. LIS was created from The "Laboratoire Fondamentale de Marseille" (LIF) in conjunction with the "Laboratoire des Sciences de l'Information et des Systèmes (LSIS).

1.2 Supervisors and partner of the LIS

LIS is a research lab which is put under the supervision of the "Centre National de la Recherche Scientfique" (CNRS), Aix-Marseille University and University of Toulon.

The LIS also has as a partener the "Ecole Centrale de Marseille" (ECM).

1.3 Localization of the laboratory

The Computer Science and Systems Laboratory is located in France and is split upon two cities, Marseille and Toulon. On Marseille we can find two places that are used by the LIS: the first one being on the university campus of Saint-Jérôme and the other on Luminy. Also on Toulon the campus of the University of Toulon is where the LIS operates.

1.4 Workforce

Spread over all of theses locations, the LIS is represented by 375 members from which we can count:

- 190 Tenure researchers and professors
- 125 Doctoral students
- 40 Post-docs
- 20 Technical staff

1.5 Reason behind the creation of the LIS

The Computer Science and Systems Laboratory was created so that we can use the strengths and competences of both LSIS and LIF together in order to meet and overcome new scientific challenges.

1.6 Hierarchy of the laboratory

While keeping this objective in head this laboratory created four departments. Each of them will inherit and focus their research on a specific domain that was previously studied by the LSIS, the LIF or both of them. Those four departments are:

- Calculus
- Data science

- Signal and image
- System Analysis and Automatic Control

Also each department is divided into subgroups working on specific topic treated by their department.

1.6.1 Calculus department

This department of the LIS was created with the objective to focus on some specific research activities like: Theoretical computer science, logic, obviously algorithmic and complexity, geometric and topology, but also quantum computing and artificial intelligence.

My internship was done with a subgroup of this department in Marseille - Luminy.

1.6.2 Data science department

Nowadays we know the importance of data, should it be founding it, treating data or even analyzing it.

So at the LIS a department is dedicated to the science of data, they are working on machine learning which is a growing technology, natural language processing that is also an upcoming technology that will be vastly used, data mining and information retrieval to collect data which is as important as computing it, and of course artificial intelligence that work hand in hand with machine learning.

1.6.3 Signal and image department

The art of manipulating visual data is represented by this department. As showing information is as important as gathering it a whole department is dedicated to this.

They especially work on image processing to treat images an recover data from them. But also on audio and bio-signal processing, medical imaging and here we see the importance of this department along the fact that they work with the medical field. And lastly image modeling is a topic that this department is competent to treat.

1.6.4 System Analysis and Automatic Control (ACS) department

The last department to be presented is making research on control theory, diagnostic, decision theory, system simulation and modeling too.

This department is focus on the analysis of systems, their correctness and also the automatic controls that can be made over them.

Chapter 2

Subject matter

2.1 Introduction to subject matter

In this internship we will try to understand the properties of a minimal Manhattan network and try to calculate it upon some given data.
For that we need to firstly describe what is a Manhattan network.

2.1.1 Manhattan network

A Manhattan network is defined over a set of points in a x dimensional space. With x going from 2 to $+\infty$ (Manhattan network on a 1 dimensional space is trivial: is just a line going through all points).

Of course the more dimensions are defining your space, the harder is the problem.

In our internship we will stick with $\mathbb{R}^2$ as a space to work on.

To create a Manhattan network the rules are simple:

1. Define the space we are working in (number of dimensions)
2. Give a cloud of points(terminals) on this space
3. Use *unit coordinates* to simplify computations
4. Calculate the Pareto envelope
5. Calculate generating set over the terminals
6. Use approximating algorithms and rounding algorithms to compute the MMN

In this report we will guide you to understand all words and steps described in these instructions.

To have an quick description of a Manhattan network you just need to know that, over a given set of points in a x dimensional space, we will have to connect every possible pairs of points with edges. Those edges can only be in two directions.

For example in the normed plane the edges are either vertical either horizontal, you can see an example just below in Fig.2.1 .

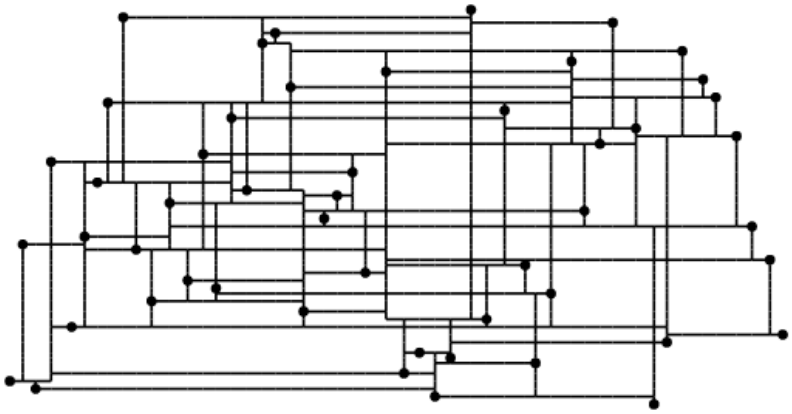

Figure 2.1: A minimal Manhattan network.

But in general any two directions can define the edge's possible layouts for our network.

2.1.2 Minimal Manhattan networks

A Minimal Manhattan network has obviously the same properties as a Manhattan network, but we add to them a constraint: the sum of the length of all edges (lines in the network) must be the lowest possible.

It is by using numerous approximating and/or rounding algorithms and data structures that we will ensure that what we create is a real minimal Manhattan network.

This simple constraint will totally change the way we approach the problem and how we shall solve it. By trying to find a minimal solution we are greatly increasing the complexity of the work that will be needed to do to answer our question.

2.2 First paper publish on the topic

Gudmundsson, Levcopoulos and Narasimhan are three scientists that worked together in the 90's on this specific topic. Finding a Minimal Manhattan network for a given set of terminals as quick as possible was their objective. In fact at the time it wasn't known that this problem was actually NP-Hard, and required a lot of calculation to be solved.

This is why after a long work the pioneers of the MMN published two algorithms with factors 4 and 8. Knowing that this problem was bind to be difficult they emitted the conjecture that a factor 2 algorithm must exist.

And it is at this time that they opened the door to a vast community of researchers to work on the topic and also variants of this problem.

But before going deeper into the explanation we need to define the elements that we are going to work on.

2.3 Origins of the topic

The minimal Manhattan network problem was in fact formulated in 1999 by Gudmundsson, Levcopoulos and Narasimhan whom found the two approximating algorithms.

Moreover they did conjectured the existence of an approximating algorithm of factor 2, without having any idea if they were close to find this algorithm or not. Nevertheless that this conjectured algorithm could give the solution to our problem with a cost lower than two times the optimal cost to find the optimal solution.

Actually, this algorithm was found in 2005 by my internship's mentors, Y.Vaxes and V.Chepoi, along with K.Nouioua and N.Catusse by using rounding method on a factional optimal solution, and then coupling it with a primal-dual algorithm.

Later other variants of this problem were studied, like for example the generalization of the problem to the normed plane with polygonal balls (in the l1-metric this polygon ball is a lozenge and not a always a square). And for this version of the MMN an algorithm factor 2.5 was found [2].

Part II

Calculate a minimal Manhattan network

Chapter 3

Terminals

3.1 Points in the space

Points in the space are named **terminals**, in this paper a terminal will refer to a point in the space.

For our study, we will focus ourselves on two dimensional space, but keep in mind that with a little more work and a lot of adjustments, all of what you are going to discover can be applied to more dimensions.

A point will be defined with Cartesian coordinates. In this paper we will note terminals with lower case letters. We will talk of a point $\boldsymbol{p}$ or a point $\boldsymbol{q}$ for example.

When speaking about one of the coordinate of a point we will use subscript notation as: x_p for the x coordinate of the point p.

3.2 Terminal's properties

Before going further in the subject, we will need to define some properties of our terminals.

These properties are the core of our computation as we will use them in order to run some algorithms over the set of terminals. As it is of high importance, the definition of these properties are given here.

3.2.1 Domination

Let T bet a cloud of points in two dimensions.

Let p and q be two terminals of T.

It is said that p dominates q if and only if:

1. $\forall t \in T : d(p,t) \leq d(q,t)$
2. $\exists t' \in T : d(p,t) < d(q,t)$

With d(p,t) the distance between p and t, that is calculated based on the metric that we use. The next whole chapter is dedicated to, and will explain, metrics and how do we calculate distances.

In this definition we need to note that no point can be dominated. As for all t in T d(p,t) must be lower or equal to d(q,t), (of course we take in consideration that p and q are different points) then t could be q. In this case then it is impossible to have $d(p,q) \leq d(q,q)$.

In any case if a terminal p dominates another terminal q we denote this property by: $p \succ q$

Moreover domination is a partial order relation, this property will be used later as we will need it to treat simplified data but equivalent to our original cloud of terminals in order to make computation faster (this simplification is covered in the unit coordinate chapter).

3.2.2 Efficiency

A point p is said to be efficient if $\neg\exists q$ and q dominates p. In other words: a terminal is said efficient if no other terminal is dominating it.

As no points should be dominated, in fact all terminals in our cloud of points are efficient terminals.

3.2.3 Equivalence

Let A be a subset of our space $\mathbb{R}^m$: $A \subset \mathbb{R}^m$.

Let T be a cloud of points in $\mathbb{R}^m$: $T \in \mathbb{R}^m$.

Let p and q be two points of T: $p \in T$ and $q \in T$.

Our two points, p and q are said to be equivalent by A if and only if:

- $\forall a \in A : d(p, a) = d(q, a)$

So as we see here, for a part of our space we can define two points as equivalent if and only if for all terminals in this part, our two points are equidistant to each and every point in this part.

We denote this property as: $p \simeq_A q$ if p and q are equivalent by A.

3.3 A set of terminals to begin with

To work on finding a MMN, we have to understand on what data we are searching this network. The objective of such a network is to **connect each an every possible pairs of points** in a set of terminals (in our case on the plane).

In fact connecting all possible pairs of terminals given a list of terminals is not difficult at all (for example just create the complete grid of your set of terminals:

Algorithm 1: Complete grid from a set of Terminals "T"

Result: The complete grid of a set of terminals in 2D
lowX = lowest x value of all terminals;
highX = highest x value of all terminals;
lowY = lowest y value of all terminals;
highY = highest y value of all terminals;
completeGrid = [An empty list of lines];
for *Every terminal t in our set of terminals* **do**
    Add to completeGrid the line from (lowX,y_t) to (highX,y_t);
    Add to completeGrid the line from (x_t,lowY) to (x_t,highY);
end
return completeGrid;

Here we can see that collecting the minimal and maximal x and y coordinates, is a linear operation and the creation of both lines are also linear operations. In conclusion a naive Manhattan network can be found in O(n).

However, generally this naive answer to our problem is far from the minimal Manhattan network. And here is the trick, this is what we want a ***Minimal*** Manhattan network.

Chapter 4

Metrics

4.1 Definition

A metric is a function, more precisely a distance function. It is used to define how do we calculate the distance between two elements in a set.

This function takes as input two elements from a set and outputs the distance between them. Of course their are many ways to calculate a distance between two elements based on what you want and what type of data you are working on.

For a classic presentation we will introduce the l2-metric. Later in this report we will use the l1-metric as it is the one that is important for us.

4.2 Example of metrics

4.2.1 The l2-metric

For the sake of simplification we will just present you the l2-metric, which is the one that we use in the "all days" world.

In fact, the **l2-metric** is also know as the **Euclidean distance** and uses the formula relative to the Pythagorean theorem:

$$d_2(p,q) = d_2(q,p) = ||p-q|| = \sqrt{(x_q - x_p)^2 + (y_q - y_p)^2}$$

This is the casual way to calculate distances on a map or between points.

4.2.2 The l1-metric

And now as simple as it is, the **l1-metric** is just another way to calculate the distance between two points p and q:

$$d_1(p,q) = d_1(q,p) = |x_p - x_q| + |y_p - y_q|$$

What is interesting in this metric here, is that we calculate a distance based of the variation of x coordinates and the variation of y coordinates.

Which means that we give some importance to the fact that we can only create edges on those two axes. This will be really useful later when we will speak about Pareto envelopes and minimal Manhattan network in more details.

Chapter 5

Interval with triangular equality

Before jumping in and defining what is the Pareto envelope of our set of terminals, or even saying why will we calculated it, we need to define the interval of points that respect the triangular equality.

5.1 Triangular inequality/equality

The Triangula **in**equality states that:

`For a triangle the sum of the length of any two sides of this triangle, must be greater than the length of the third side.`

Which means that: let abc be a triangle:

- $d(a,b) + d(b,c) > d(a,c)$
- $d(a,c) + d(c,b) > d(a,b)$
- $d(b,a) + d(a,c) > d(b,c)$

Now take in consideration two point a and b, all c points that respect the triangular equality are the ones that do not verify the triangular inequality which means that:

- $d(a,c) + d(c,b) \leq d(a,b)$

5.2 Interval and metrics

But as we defined previously d(x,y), we know that this notation depends on the metric that we use. Moreover, depending on which metric we use the set of points that will respect the triangular equality will differ. Here are some examples of this set depending on the metric used:

$I_{d_1}(x,y)$ $I_{d_2}(x,y)$

Figure 5.1: Triangular equality interval depending on the metric.

As you can see on the right, this set is just all the points on a straight line between x and y. In fact as we use the l2-metric (Cartesian distance) we fall back on our feet with the casual definition of the triangular inequality.

Nevertheless, on the left we see that for the l1-metric this interval is a rectangle, with the segment [x,y] being a diagonal of this square. And you can see that after doing the calculation we actually end up with this shape for all points that respect the triangular equality.

5.3 Usage of intervals

Also as we will work a bit with the l1-metric, this result will be really useful for future computations.

All the points that respect the triangular equality can be regrouped in a set. This set will simply be called interval between x and y for the metric d_1(resp. $d_{metric\_identifier}$).

It is base on this interval that we will also define the Pareto envelope of our set of terminals.

Note: As the construction of the Pareto envelope can be linked to the interval between all pair of points, and as this said interval depends on the metric that we use to calculate distances, **the Pareto envelope of a set of terminals depends on the metric that we use**.

5.4 Used notation

By defining this interval we now can have a way to start and work our way to the Pareto envelope of our set.

This interval will be noted as: I_d.
With d the identifier of the metric used to calculate this interval.

Chapter 6

Unit coordinates

6.1 What are unit coordinates

Granted that we are working in a two dimensional space, all coordinates are more likely to be real numbers than natural numbers. Moreover our system would be useless if we were only working on discrete coordinates, where our world and the way we measure distance is not discrete.

However we can counter this by having a tool or an algorithm that can convert two dimensional coordinates into discrete coordinates. Unit coordinates are a way to locate points in space with only round numbers.

Additionally this will restrict the space we are working on from $\mathbb{R}^2$ to a grid containing all of our terminals.

This tool would greatly lower our computation time and will make our future algorithm more easier to implement.

6.2 Why do we use them

Under those circumstances, as said it make implementation of future algorithm simpler as we are standardizing the way we are talking about coordinates and points. As a result all future algorithm will share the same references to coordinates and it will be simpler to consecutively run algorithms on the same data.

Furthermore speaking of data, this will also define the data structure used to store our terminals. In my own implementation terminals in $\mathbb{R}^2$ are read from a CSV file and stored as a list. Just after that I use an algorithm designed by myself to convert those real coordinates into unit coordinate.

6.3 Recovering original coordinates after computations

Despite having unit coordinates making our job easier, we need at the end of all our computation to be able to transition back to the real points in $\mathbb{R}^2$ because otherwise using unit coordinates would not make sens.

For this we created a specific data structure (in my implementation a python dictionary) to link each real point to his unit coordinate translation by our algorithm.

After having found a minimal Manhattan network for our set of unit coordinates points, we will use the fact that in the view of Pareto envelope and Manhattan network (and it has to work for both as the first is used to create the second), unit coordinate is an **isotone mapping** of our points.

Meaning that for a partial order relation $\boldsymbol{R}$, applying an isotone mapping **will not** change the order of any elements in $\boldsymbol{R}$.

And as a matter of fact we have to remember ourselves that terminal domination is a partial order relation that is used to define all of the structure we are talking about.

If we keep this relation untouched, we just shown that if we found a MMN for our unit coordinate grid, it is the same thing than finding a MMN for the set for terminals in $\mathbb{R}^2$. We will just have to convert our point back to what they were in $\mathbb{R}^2$.

Chapter 7

Pareto envelopes

7.1 Definition of Pareto envelopes

7.1.1 Simple definition

The Pareto envelope of a set of terminals is a structure that is defined by a subset of terminals constructing it. This subset is formed with only all the non-dominated points in the space.

7.1.2 Remarkable result

If we study the the Pareto envelope and the convex hull(also known as convex envelope) of a cloud of points we see that they coincide but only if we use the euclidean distance (l2-metric).

This result is important as it make us understand that **convex envelopes** and **Pareto envelopes** are some kind of linked but **not the same object**.

7.2 Calculation with I_d in $\mathbb{R}^2$

7.2.1 A set of efficient points: Υ_d

Before calculating the Pareto envelope we need to be introduced, referring to a set of terminals, to the set of efficient points that is linked to it.

This set is determined by the intervals that we calculated previously. So it is natural that this set is totally depending on the metric we are working with.

It is in that matter that we have to know that the Pareto envelope depends also on the metric used to calculate distances.

7.2.2 Construction of Υ_d

This set is noted Υ_d and is defined as:

$$\Upsilon_d = \cap_{i=1}^{n}(\cup_{j=1}^{n} I_d(t_i, t_j)) \tag{7.1}$$

As you can see the equation 7.1 stipulates that the set of efficient points is defined by the intersection of all intervals from all points to all others. Meaning that Υ_d is defined by the intersection of all intervals with all pairs of points possible.

And it is with this definition that we see that we are getting closer and closer to construct our target, a minimal Manhattan network.

7.2.3 Υ_d in $\mathbb{R}^2$

With this tool we can now construct the set of efficient point in our cloud of points in $\mathbb{R}^m$.

As we are for now working a specific space ($\mathbb{R}^2$), we can now specifically construct Υ_{d_1} for the l1-metric and Υ_{d_∞} in the L-infinity metric.

7.3 Different behavior in $\mathbb{R}^3$

This remark is here for the continuation of this intern ship and will be useful for later.

We have to be careful when working on higher dimension and to not generalize our result too quickly.

In fact as for $\mathbb{R}^2$, the Pareto envelope is Υ_{d_1} or Υ_{d_∞} depending on the metric used, it is not the case in $\mathbb{R}^3$.

In $\mathbb{R}^3$ the Pareto envelope is more complex to calculate and moreover differs from Υ_d shall it be with euclidean distances, l1-metric or even l-infinite metric.

As the Pareto envelope is calculated in $O(nlog(n))$ in $\mathbb{R}^2$ due to this difference in $\mathbb{R}^3$, the Pareto envelope in $\mathbb{R}^3$ is not that easy to compute.

Again as we are working in $\mathbb{R}^2$ we are going to discover an algorithm designed to calculate the Pareto envelope in such a space in $O(nlog(n))$ as it is the lowest complexity to do so.

7.4 Construction of the Pareto envelope with a scanning algorithm

7.4.1 Dividing the problem into chains

The easiest way to construct the Pareto envelope is to divide the problem into four chains. For that we will construct the chains $S1$,$S2$,$S3$,$S4$. Clockwise the chains are built in this manner:

- $S1$ - North to East
- $S2$ - East to South
- $S3$ - South to West
- $S4$ - West to North

After computing all four chains, concatenate them in the right order $S1 -> S4$ and this data structure will represent the Pareto envelope.

7.4.2 Computing the chains

Here is an algorithm to calculate the first of the four chains.

Algorithm 2: Compute chain S1

Result: The S1 chain from a could of terminals "T"
Sort the terminals by decreasing ordinate and increasing abscissa.;
Let T be the set of terminals.(T = [t_0, t_1, ..., t_{n-1}]);
S1 = ∅;
previous = t_0;
for *i=2 to (n-1)* **do**
  if $x_{t_i} > x_{previous}$ *AND* $y_{t_i} == y_{previous}$ **then**
    $S1 = S1 \cup [previous, t_i]$;
    $previous = t_i$;
  if $x_{t_i} > x_{previous}$ *and* $y_{t_i} > y_{previous}$ **then**
    temp = $(x_{previous}, y_{t_i})$
    $S1 = S1 \cup [previous, temp] \cup [temp, t_i]$;
    $previous = t_i$;
end
return S1;

Here S1 is a list of points, these points represent the "breaking points" along the chain. Drawing this chain can be done by drawing a line from a point to his successor in the list.

Remark: The construction of S2,S3 and S4 is done in the same way, the only thing to change are:

1. The sort applied to T.
2. The conditions in the two "if" statements.
3. The name of the returned variable.

7.4.3 Complexity

Despite doing this algorithm four time, the overall complexity (construction of the four chains) is $O(nlog(n))$.

The first step in this algorithm, is to sort the points. Obviously we use an $O(nlog(n))$ algorithm for that as it is the quickest way to do so.

All other instruction are made in linear time. We only insert points into a list which is a linear timely done and verify condition (also in linear time).

- Line 1 // Sorting the terminals // $O(nlog(n))$
- Line 2 to 4 // Initialization // $O(1)$
- Line 5 to 13 // For loop // $O(n)$

The complexity of this algorithm is:

$O(nlog(n)) * O(1) * O(n) = O(nlog(n))$

Chapter 8

Minimal Manhattan network (MMN)

At this stage, most of the work is already done, we just only miss the computation of the minimal Manhattan network.

Clearly, with all the data we have now from our set it will be (with the usage of some tricks) not that hard to compute it.

Now that we have the Pareto envelope of our set, we just need to find a generating set over our terminals and satisfy it.

In this chapter we will cover what is a generating set and why satisfying such a set is equivalent to finding a minimal Manhattan network.

8.1 Generating set

Let A be a generating set over a cloud of terminals T.
A respects the following assertions:

1. A is a subset of T: $A \subseteq T$
2. Connecting all pairs of A **implies** All pairs of T are connected

As a matter of fact, we now know that we can in order to find a minimal Manhattan network, try to find a generating set over our terminals.

By studying two specific geometrical shapes in our set T we will be able to create such a generating set, thus solving our problem: finding a minimal Manhattan network.

8.2 Terminal's neighbours

As we are trying to find geometric shapes that can be defined as a generating set, we need to be able to talk about the neighbours of a point.

In fact we will create appropriated data structures so that we keep in memory the neighbours of each point.

8.2.1 Q_i, A_i, and O_i dictionaries.

For that we will use dictionaries:

- Step one: As you can see in *Fig 8.1* is for a point to divide the space around it in four squares. $(Q_i \mid \forall i \in [1, 4])$
- Step two: For each Q_i we will create the appropriated A_i and O_i

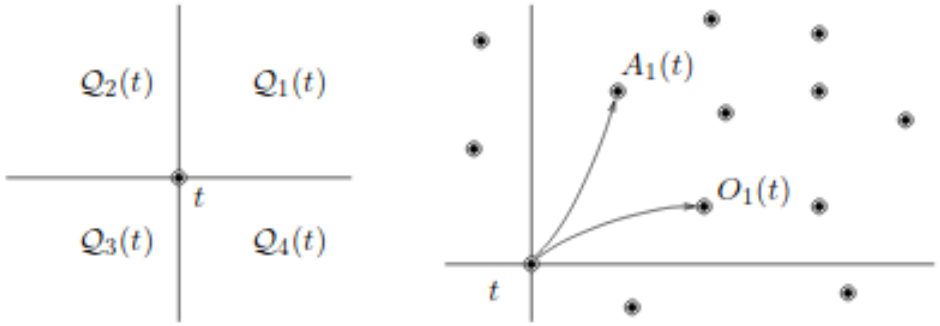

Figure 8.1: Terminal's neighbours

8.2.2 Definition of A_i and O_i

Let p and q be two point in T. ($p \in T$ and $q \in T$)
Let A_i be linked to Q_i. (i.e A_1 is linked to Q_1 for example)
Let A_i be a dictionary from a point to another (i.e $A_i(point1) = point2$)

$A_i(p) = q$ means that:

- $q \in Q_i$
- q is the closest point to the a **vertical line** going through p

$O_i(p) = q$ means that:

- $q \in Q_i$
- q is the closest point to the an **horizontal line** going through p

8.2.3 Algorithm to create A_i and O_i

In this part I will present you an algorithm to generate O_1.
Note: All other dictionaries can be created similarly just by giving small modifications on the following algorithm.

Algorithm 3: Calculate O_1

Result: The 0_1 dictionary properly filled
Sort T by increasing ordinates and increasing abscissa;
O_1 = An empty dictionary;
toTreat = An empty stack;
$toTreat.push(t_0)$;
for *i=1 to n* **do**
  while *First element of toTreat has an abscissa* $\leq x_{t_i}$ **do**
    s = toTreat.pop();
    $O_1(s) = t_i$;
  end
  toTreat.push(t_i);
end
while *The stack toTreat is not empty* **do**
  s = toTreat.pop();
  $O_1(s) = \emptyset$;
end
return O_1

8.3 Strips

In this section we will cover strips which are composed with the set of vertical and horizontal strips created by our points.

8.3.1 What are strips

Vertical(reps. Horizontal) strips are geometrical forms made by two points that are successive in y(reps. x) coordinate.

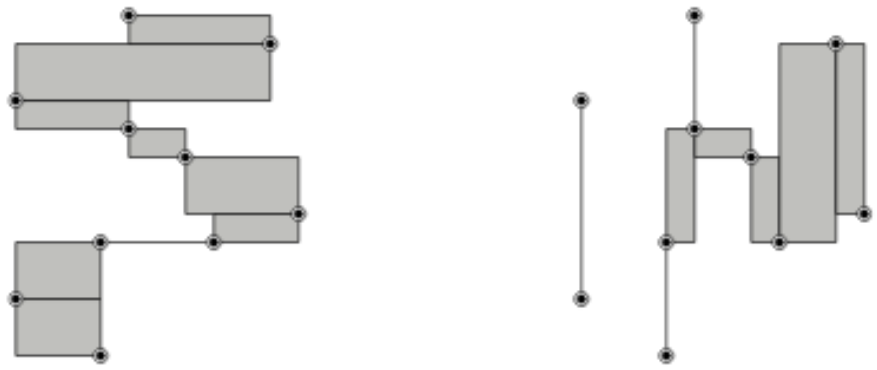

Figure 8.2: Vertical and horizontal strips.

8.3.2 Algorithm to locate strips

Algorithm 4: LocateStrip

Result: HS and VS as lists of tuples. (respectively horizontal strips and vertical strips

HS = ø;
VS = ø;
Calculate O_i and $A_i \mid \forall i \in [1, 4]$;
for *i=1 to n* **do**
  if $t_i = O_3(O_1(t_i))$ **then**
    $HS = HS \cup (t_i, O_1(t_i))$;
  if $t_i = O_4(O_2(t_i))$ **then**
    $HS = HS \cup (t_i, O_2(t_i))$;
  if $t_i = A_3(A_1(t_i))$ **then**
    $VS = VS \cup (t_i, A_1(t_i))$;
  if $t_i = A_4(A_2(t_i))$ **then**
    $VS = VS \cup (t_i, A_2(t_i))$;
end
return HS and VS;

8.4 Staircases

Staircases are another geometric shape with a specific configuration.

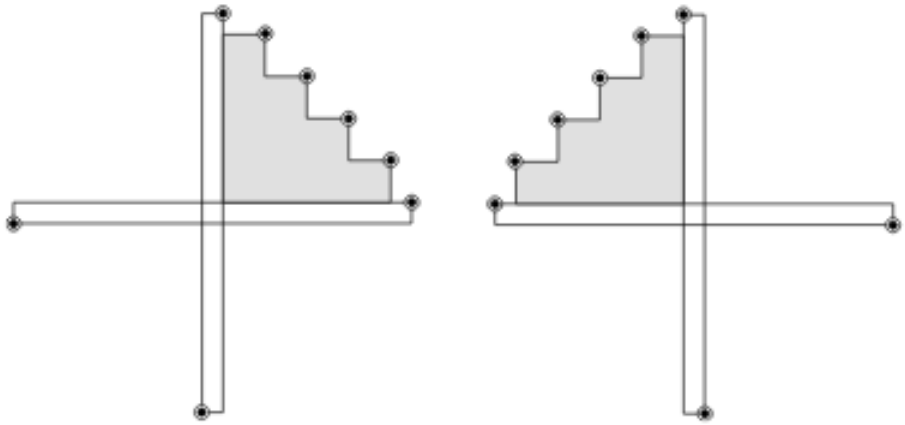

Figure 8.3: Staircases configuration.

Here above on *Fig 8.3* are presented two of the four possible staircases configuration.

As you can see a staircase is built on two strips, one vertical and the other one horizontal. Also note the position of the points on the strips. The points must be on a disposition where the points on the staircase side are part of the staircase itself.

8.5 The resulting set

Now that we have strips and staircases we can create a generating set.

In fact all strips combined with all staircases are a generating set for our cloud of points.

Meaning that at this point we can now with the data that we have, instead of treating all terminals we can just focus on our staircases and strips.

All of the done work reaches this point. What is missing for me is to properly locate staircases, and implement an approximating algorithm described [2] to calculate a minimal Manhattan network.

Part III

Conclusion and annexes

Chapter 9

Internship proceeding

This chapter is here to summarize quickly how did the internship went on, and what were the encountered difficulties.

9.1 Covid-19 situation

Due to the current situation the internship was made from my own home, as the confinement restriction didn't allowed me to work directly at the laboratory or at any library.

In this respect, the internship was for me a little harder as communicating with my internship mentors via e-mails was slower than asking direct questions to them. However they never missed on answering my questions and with a really helpful accuracy.

Unfortunately, due to this situation the internship was slowed and it was harder for me to work. I personally think that working from home is fine but not optimal, as the environment is not meant for working is would rather have worked directly at the laboratory to be more productive and efficient.

I hope that to a certain extent, you understand my position and take in consideration the fact that my work was greatly slowed by this situation.

9.2 Encountered difficulties

Most of the difficulties that I encountered was the consequence of the lack of knowledge on some specific topic. I was just stuck sometimes in my work, while calculating the Pareto envelope. Hopefully after a few mails to my internship's mentors, they redirected me to research papers and publication where all of what I needed was explained.

Also during video-conferences they explained me a lot of the properties hold by Pareto envelopes, strips and staircases.

However during the internship we had a lot of final exams and numerous other semester's projects to submit to our professors.

This took a lot of the work time that I could have added to this internship. A lot of work had to be done in the same as this internship and managing our time was really hard for us. I hope that in view of this situation you understand that I did my best and produced the best work I could for this internship while also trying to get the best marks by additionally giving the best of me for the other projects and exams too.

9.3 Current status of the work

In terms of code, my strategy was to code by implementing tests first. This technique was tough to me last year and it proved it's effectiveness again here.

As a consequence coding all tests before implementing my own code was a big guideline for me to know where I am in my work. This internship also tough me how to organize myself in the regard of a project in order to have clear and neat code.

My aim was to reach a point where i can compute a minimal Manhattan network, which is on a good way to be done. For now i did all the work needed to recognize Pareto envelope and to get the strips and staircases set.

The last part of my work which will be done in the next three weeks is to initiate myself to linear programming (for personal knowledge), and lastly in order to compute a MMN I need to implement a 2.5 approximating algorithm that does it([2]).

I was hopping to finish all of my work before presenting you this report so that I could completely explain all of my work, but due to the current situation this was too hard to do. Hopefully my internship is giving me, I think, enough time to do the rest of my work.

9.4 Continuation of the internship

Finally i have to say that my internship is not finished and I will continue to work from home during approximately three to four weeks which will be enough to finish my work on this internship.

As this report is a manifest of the end of the matter "Stage ou projet - S6", I would like to thank some people before ending this report.

Chapter 10

Acknowledgment

In this last chapter I would like firstly to thank you for reading this report. I hope that this was accurate enough and that this was constructing for you. For this internship I have to thank a lot of people that helped me through this adventure.

10.1 Internship's mentors

With all the work i produced, I would like to thank Mr.Chepoi and Mr.Vaxes my intership's mentors, who despite the current covid-19 situation helped me.

With video-conferences and mail exchanges they managed to keep me interested in research and especially in this subject matter. I cannot count the answers that they gave to the millions of questions that i had and always with a pedagogic approach.

Even though this internship's subject was hard they always pushed me to try and understand, search, and learn what i needed to know to move forward.

In this confinement circumstances, while home working I didn't felt alone at any time. When I had questions, my internship's mentor gave me enough information so that I could work by myself. The objective of this internship was to make me learn more about research, ho it is done and of course learn more about Pareto envelopes and Manhattan networks.

I personally feel that I did even more than that, I had to learn how to read scientific publications, how to search information in a bibliography, how to retrieve from Internet publish articles, understand how do some data structures works without knowing them before. All of this without forgetting that I made a lot of discoveries.

Additionally I had to prepare small talks for my video-conferences with Mr.Vaxes and Mr.Chepoi, to explain them what did I learn, what did I do, and what were my questions. I learned a lot on paper writing also.

10.2 Making this internship possible

Preparing all of the required papers for an internship is compulsory and a long work. And I cannot end this paper without showing how much grateful I am to three particular persons.

Especially in the current sanitation situation, retrieving all the needed papers, and all needed signatures was not a soft task to do. Alone i would never have been able to do so.

For this I have to gratefully thank Mme.JAMET and Mme.JEANSON whom helped me a lot with the paperwork for the University side.I took

some time to understand what was wanted from me. But after starting the procedures, and mails after mails Mme.JEANSON and Mme.JAMET always answered me with astonishing rapidity and accurate instructions. And for this thank again.

For the laboratory side, I need to thank Mmme.COMES, my interlocutor for the LIS. She also help me to summaries and centralize all of my papers while starting this internship.

The current situation was not really usual but even in hard times Mme.JAMET, Mme.JEANSON and Mme.Comes helped me in making this internship possible. Nothing of this was going to be real and possible if they were not here.

Chapter 11

Bibliography and sources

Template used:

```
[Number] -{Autors}
         - Title of the scientific publication
         - Event/Congress where the paper was presented
{Links related to this source (Abstract and/or part of the paper as PDF)}
```

Sources:


```
[1] -M.Benkert, A.Wolff, F.Widmann, and T.Shirabe,
    -The minimum Manhattan network problem:approximations and exact solutions,
    -Comput.Geom.35 (2006) 188{208
pdf: https://www.sciencedirect.com/science/article/pii/S0925772105000933

[2] -N.Catusse, V.Chepoi, K.Nouioua, and Y.Vax‘es,
    -Minimum Manhattan network problem in normedplanes with polygonal balls:
     a factor 2.5 approximation algorithm,
    -Algorithmica 63 (2012) 551{567.
abstract: https://arxiv.org/abs/1004.5517
pdf:      https://arxiv.org/pdf/1004.5517.pdf

[3] -V. Chepoi, K. Nouioua, and Y. Vax‘es,
    -A rounding algorithm for approximating minimum Manhat-tan networks,
    -Theor. Comput. Sci. 390 (2008), 56{69 and APPROX-RANDOM 2005, pp. 40{51.
abstract: https://www.sciencedirect.com/science/article/pii/S0304397507007657
pdf     : Check in the link the "Download full text pdf" field.

[4] -M.Benkert, A.Wolff, F.Widmann, and T.Shirabe
    -The minimum Manhattan network problem:approximations and exact solutions,
    -Comput.Geom.35 (2006) 188{208.
https://link.springer.com/content/pdf/10.1007/s00454-011-9342-z.pdf

[5] -A.Das, K.Fleszar, S.G.Kobourov, J.Spoerhase, S.Veeramoni, A.Wolff,
    -Approximating thegeneralized minimum Manhattan network problem,
    -Algorithmica 80 (2018), 1170-1190.
abstract: https://link.springer.com/article/10.1007/s00453-017-0298-0

[6] -A.Das, E.R.Gansner, M.Kaufmann, S.G.Kobourov, J.Spoerhase, A.Wolff,
    -Approximatingminimum Manhattan networks in higher dimensions,
    -Algorithmica 71 (2015), 36{52.
abstract: https://link.springer.com/chapter/10.1007/978-3-642-23719-5_5

[7] -J.Gudmundsson, C.Levcopoulos, and G.Narasimhan,
    -Approximating a minimum Manhattan net-work.
    -Theor.Comput.Sci.390 (2008), 56{69 and APPROX-RANDOM 2005, pp.40{51.
```

https://core.ac.uk/download/pdf/82701447.pdf

[8] -K.Nouioua,
-Pareto envelopes and Manhattan networks: Algorithmic caracterisation
-Computer science Thesis, University of Mediterranee, 2005.
pdf : http://pageperso.lif.univ-mrs.fr/~karim.nouioua/download/THESE_NOUIOUA.pdf