

1.- Introduction – Overall thing to develop.

2.- Preliminaries – Basic definitions of what’s already there (TSP definition, convex hull usefullness to give another perspective)

3.- Literature review - common approach due to the relation to the question of P vs NP. Concorde, CLK, LKH, aprox methods. Libraries (TSPLib mainly).

4.- Data collection – TSPLib (explaining its consequences of having rounded up the distances when calculating results, and how it varies). Explain that we will recalculate the costs of those instances which have the tours published.

5.- Theoretical basis of PolyGenesis – All the explanation that I have. At the end, a conclusion with the pseudocode.

6.- Results – I would need to implement it and all

7.- Conclusions and future work – blah blah blah

Definifion matemática de envolvente convexa – <https://www.sciencedirect.com/topics/mathematics/convex-hull#:~:text=The%20definition%20of%20convex%20hull,smallest%20convex%20set%20containing%20X>.

The definition of convex hull is as follows: A set Y is said to be convex if for any points a, b ∈ Y, every point on the straight-line segment joining them is also in Y. The convex hull of a set of points X in Euclidean space is the smallest convex set containing X.

# Portada

# Abstract

This work presents a new method for the euclidean Traveling Salesman Problem (TSP) based on some geometrical properties of said problem. We will use the cartesian coordinates of the points that compose the instance instead of the usual graph representation, to preserve the estructural context of the instance and allow to automatically deduce the general layout of the points in the solution. To test the results we will use some of the well known datasets such as TSPLib.

# Keywords

Travelling Salesman Problem

Combinational optimization

# Index

# Introduction

# Preliminaries

In this section, key concepts in regards to TSP and geometry properties relevant for the development of the algorithm are reviewed.

## Travelling Salesman Problem - Cambiar que está cogido casi literal de Huerta

The Travelling Salesman Problem (TSP) is one of the most researched optimization problems due to its several real world applications, in tour designing, crystalography, machine production optimization, among others (………………………fuente………………..). The original problem statement is as follows: Given a set of n cities, a salesman wants to visit each one of them only, starting at any city and ending his path on the same one. The TSP looks to answer the question: Which route should the salesman take to travel the least possible distance?

We assume that the distances for any pair of cities are known and can be conceptualized as money or time. From now on, the distances will be referred to as costs. Mathematically, the problem can be stated as: Given a set of cities 1, 2, …, 𝑛 and a cost matrix 𝐶 = 𝑐𝑖𝑗, in which 𝑐𝑖𝑗 represents the cost of going from city 𝑖 to city 𝑗, with 𝑖, 𝑗 = (1, 2, …, 𝑛), find a permutation {𝑖1 , 𝑖2 , …, 𝑖𝑛 } of cities that minimizes the sum 𝑐𝑖1 𝑖2 +𝑐𝑖2 𝑖3 +𝑐𝑖3 𝑖4 + ⋯ +𝑐𝑖𝑛−1𝑖𝑛 +𝑐𝑖𝑛 𝑖1 ,where 𝑖 𝑗 𝑖 𝑗+1 represent the indexes of consecutive cities in the permutation. The properties of the cost matrix can be used to characterize the problem:

1. If 𝑐𝑖𝑗 = 𝑐𝑗𝑖 for every 𝑖 and 𝑗, the problem is symmetric , and otherwise is asymmetric.
2. If the triangular inequality is satisfied for the intercity distances (𝑐𝑖𝑘 ≤ 𝑐𝑖𝑗 + 𝑐𝑗𝑘 for every 𝑖, 𝑗 and 𝑘), then the problem is metric.
3. If 𝑐𝑖𝑗 are euclidean distances between points in the plane, the problem is euclidean. This problem is both symmetric and metric

## Convex Hull

In mathematics, the convex hull or convex envelope for a set of N points in euclidean space is the minimal convex set containing these N points (<https://ti.inf.ethz.ch/ew/courses/CG13/lecture/Chapter%203.pdf>).

This concept will be use in repeated occasions during the algorithm development since convex polygons are simpler to analyze and will allow to transform the problem itself, as it is explained below.

# Literature review

## TSP solvers

## Instance generators and libraries

# Data collection

## TSPLib

TSPLib is one of the most referenced sources for this problem. It offers instances of a variety of sizes, which is helpful to perform time measurements. Most researchers use these instances to test their algorithms - being exact or not, and to allow us to establish some comparisons, we will choose some of these instances for this purpose.

### Relevant considerations

Even though there are some proposed solutions, these may not be the optimal ones. Presumably, the source of these variations is the process of rounding up the distances between each pair of points of the instance, to lighten calculations and reduce computational costs. As an example, (incluir un ejemplo que se vea que no es exactamente la distancia y cree una diferencia)

In order to avoid imprecise comparisons, we will only use those instances for which a tour is provided, so their real cost can be calculated without rounding to units.

# Theoretical basis of the algorithm

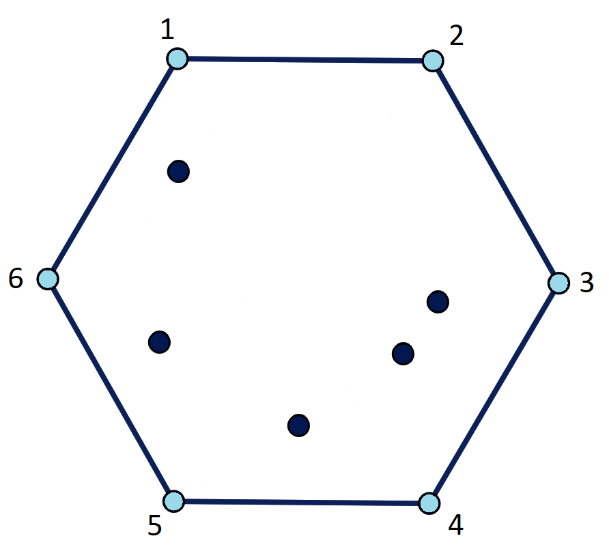
## TSP as a geometrical problem

As it was previously stated, the TSP problem's goal is to calculate the shortest path which travels through all points of a set, finishing where it started. Geometrically, if we consider this set of points as a set of vertices and the path between each pait of consecutive points as edges, the result would be a polygon of minimum perimeter. Therefore, an equivalent statement of the problem is: "Given a set of n vertices, find the set of n edges which forms a polygon of minimum perimeter".

## Convex hull and TSP

As it is a minimum perimeter polygon, another characteristic is that it must be a simple polygon. In other words, it must not be self-intersecting. This is the main fundamental of algorithms like 2-opt (https://essay.utwente.nl/72060/1/Slootbeek\_MA\_EEMCS.pdf) which are based on intersection detection.

Due to this constraint, by using the polygon's convex hull we can deduce the general layout of the end result.



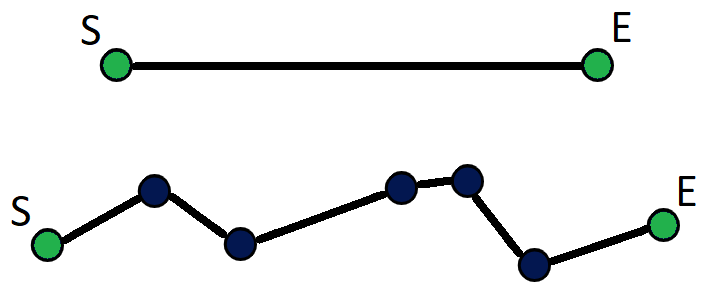
For example in this case, the blue vertices represent the convex hull. When travelling across them in order (1, 2…6 or in opposite direction, 6, 5…1) a simple polygon is obtained. However, if this order is not follow, then it will not be possible to obtain a simple polygon, therefore it would not be of minimum perimeter.

With this deduction, all instances that assembles a convex polygon have trivial solutions. But those instances which have inner points, like this previous example, would need to find the correct position of each point between any pair of consecutive points of the convex hull.

Considering a path which starts in a vertex of the convex hull and ending in the following, it is trivial to deduce that said path must be of minimum length (as if it was not the case, the whole polygon could not be of minimal perimeter) and, if it contains inner points, the path must not be self-intersecting (following the same reasoning process as above).

## Minimum Path

We will define a Minimum Path (MP) as the route of minimal length which starts in a starting point 'S', travels through a set (empty or not) of middle points and finishes in an exit point 'E'.

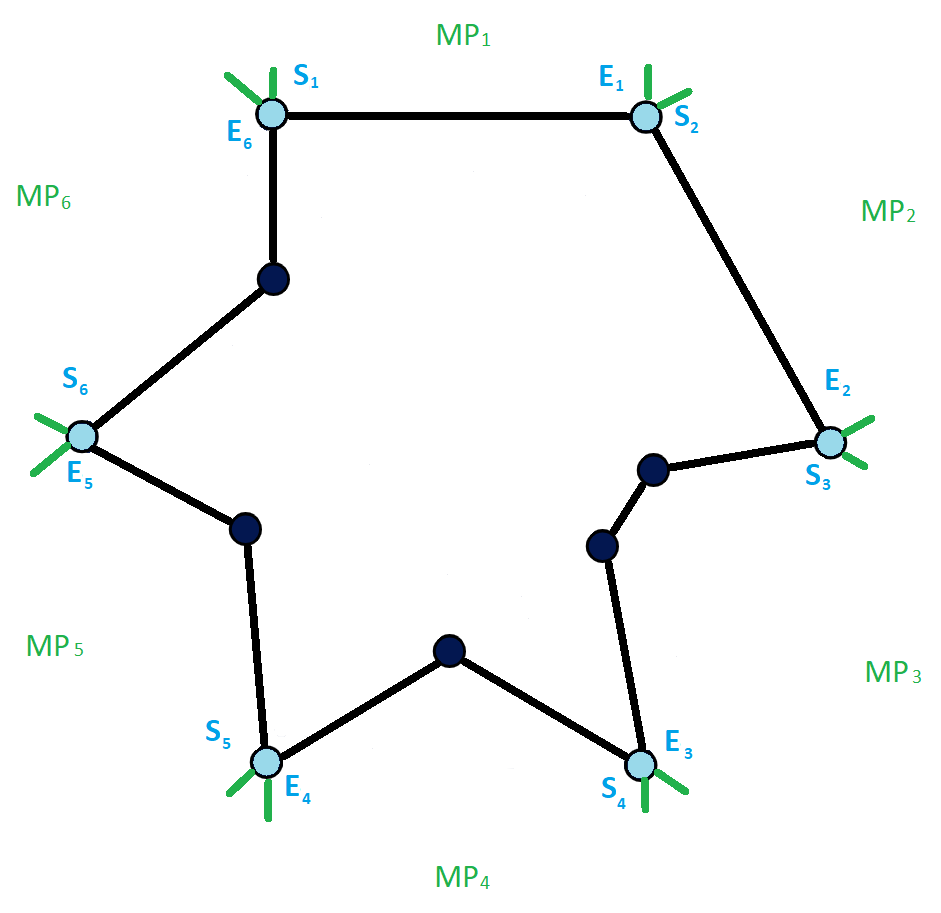


Both are examples of MPs, the first one having no middle points.

## TSP definition in terms of Minimum Paths

As a summary of the above, we can divide the TSP problem into smaller problems in regards of the minimum paths. An euclidean TSP, in terms of geometry, is a minimum perimeter polygon, by consequence being simple. To achieve simplicity it must respect its convex hull, so it can be used to divide the tour into minimum paths. These paths must follow both following characteristics:

* Their starting and exit points correspond to the consecutive points of the convex hull.
* The inner points of the polygon must be distributed across the set of minimum paths where they create the least increment in length.

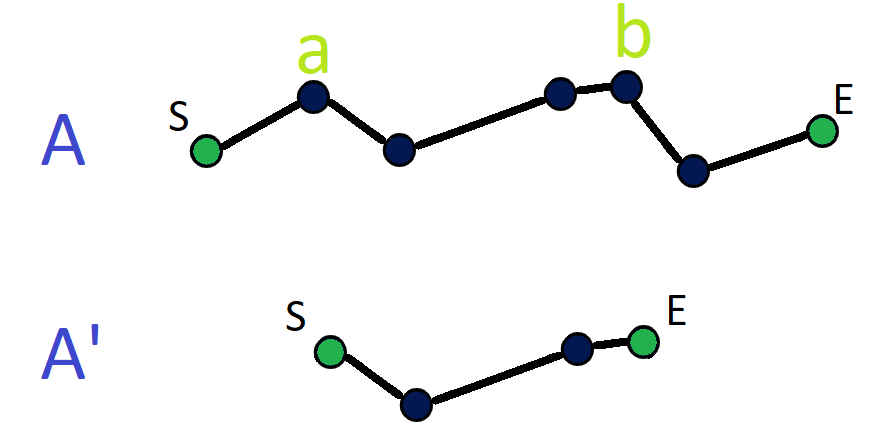


An example of a decomposition of the TSP in Minimum Paths.

To broaden the perspective to the problem, below some properties of the MPs are discussed. These properties and conclusions have the purpose to be of later use to compose our proposed algorithm.

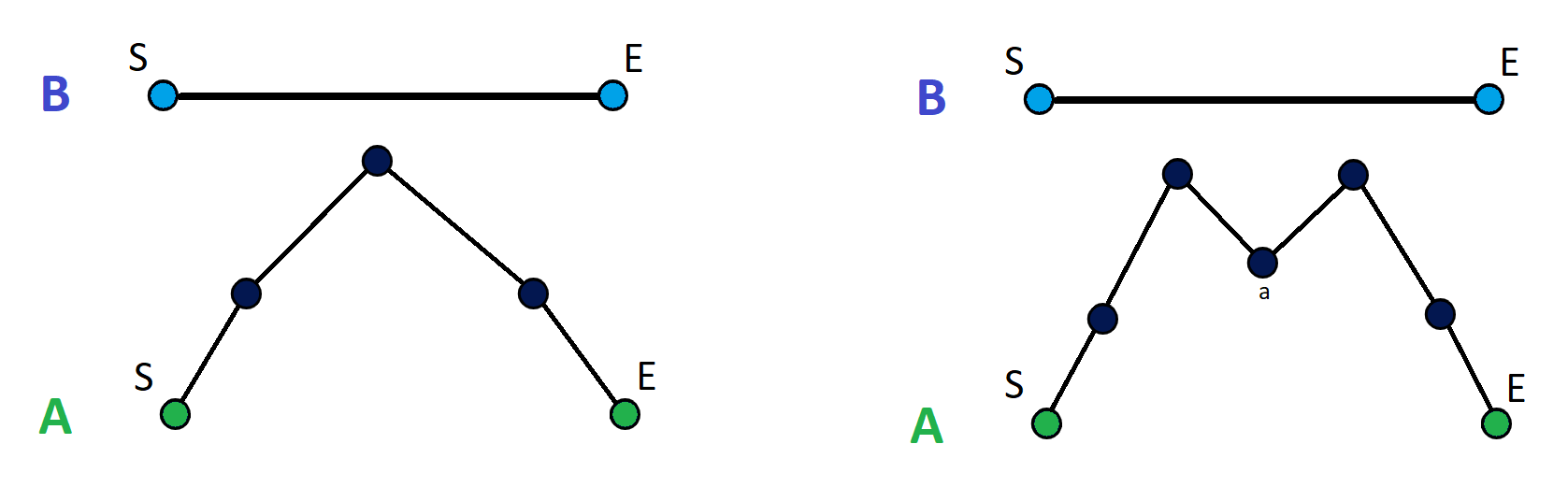
## Composition rule

A minimum path "A" is composed by other smaller minimum paths. For instance, choosing any point of A as the start point ("a") and a subsequent as exit ("b"), the selected segment (" A' ") must be a minimum path as well. If said inner segment is not minimum, "A" could not be minimum either.

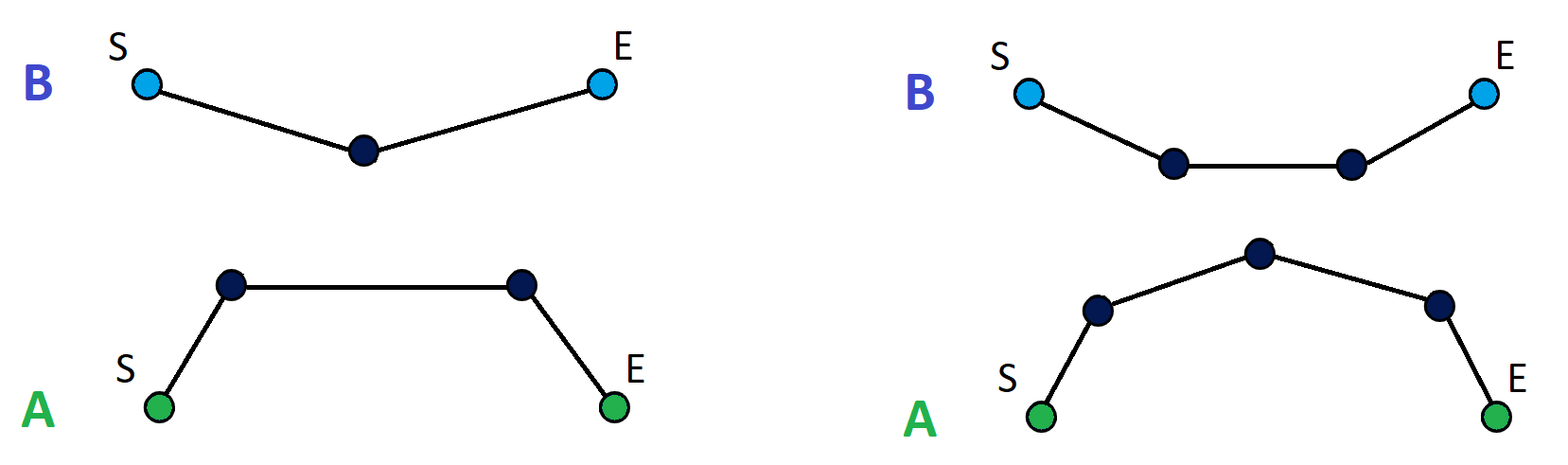


## Derivation rule - REWORK (aclarar el método siguiendo los sets de dependencia y la condicion de que si ~sECi e B => ~sECi+1 e B)

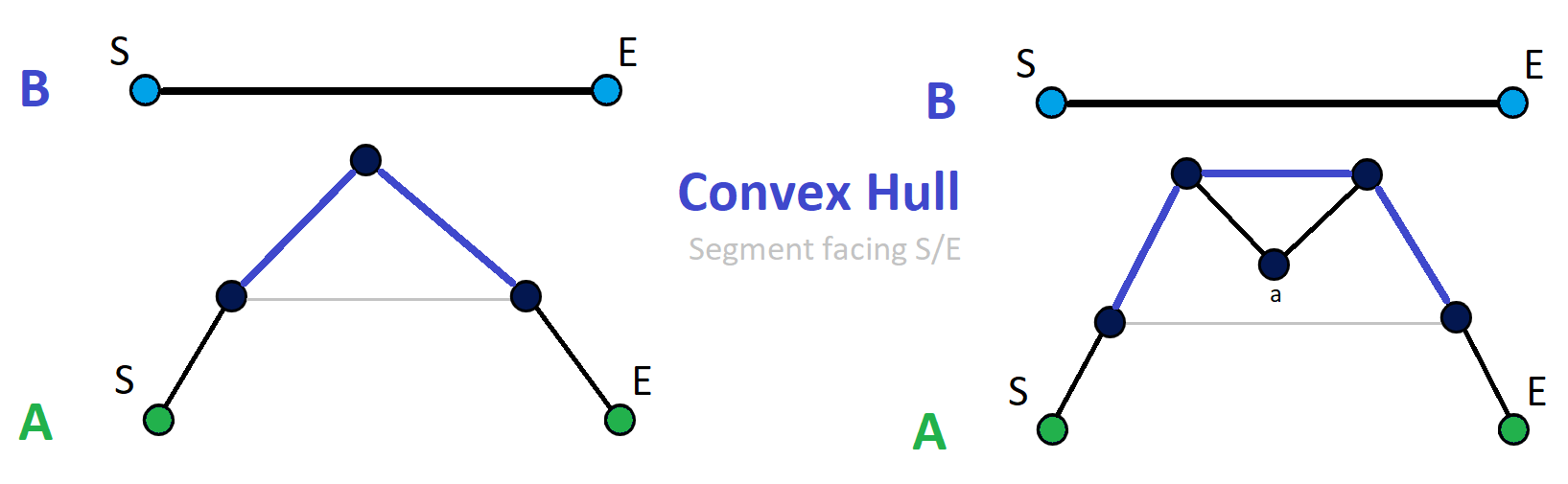
Let A and B be two minimum paths, which do not intersect. A posseses some middle points between the line linking A's start and end, and B's start and end. B does not have any middle points. To reduce the total distance of both MPs, we need to check whether there exists a set of points in A that can be moved into B to minimize the total distance. In this case the composition rule cannot be used, as there are some cases in which the set to include in B does not possess consecutive points.



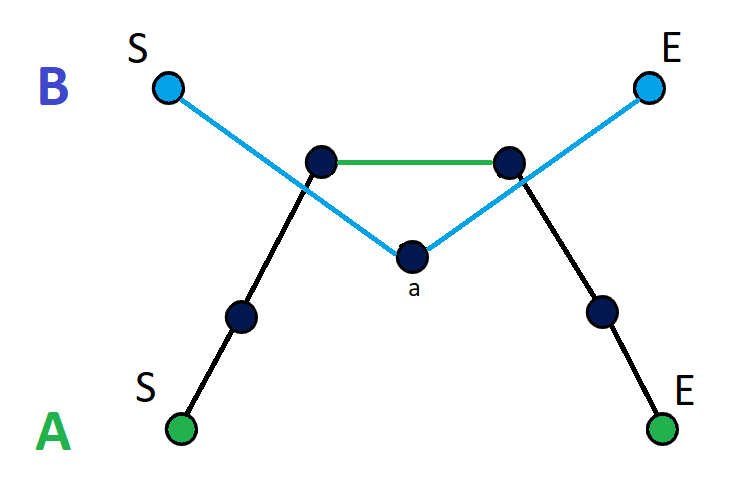
In the first case, the minimization can be performed by moving the middle point of A into B. However in the second example, the set of points to be moved includes two which are not consecutive (leaving out a middle one, 'a'). The minimization result is as follows:



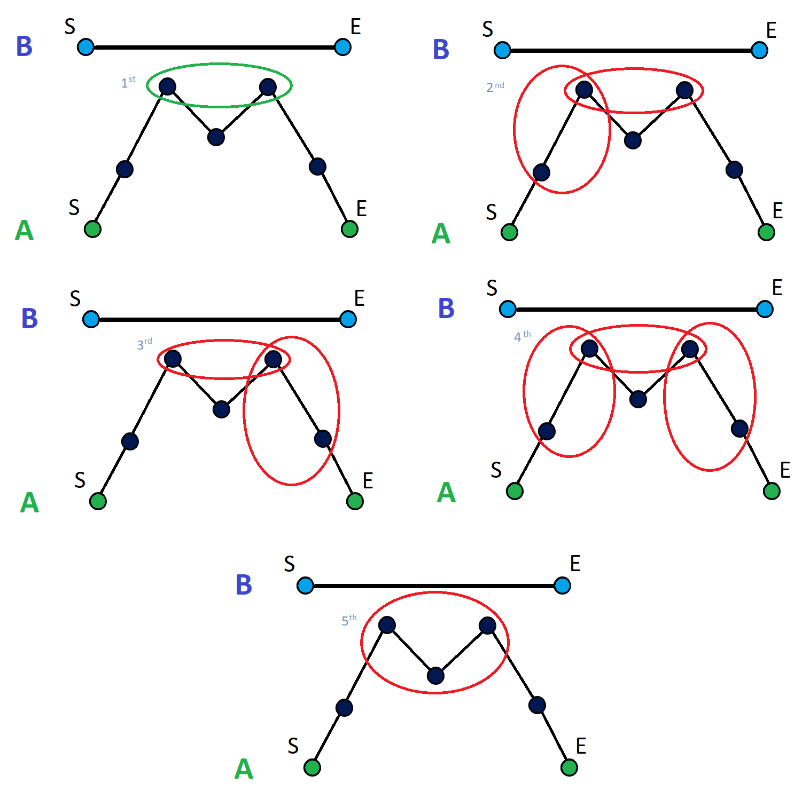
A possible method to detect both cases is using the convex hull of the middle points in A and, using its segments (with the exception of the one facing A's start and end points) check if it is appropiate to move them to B.



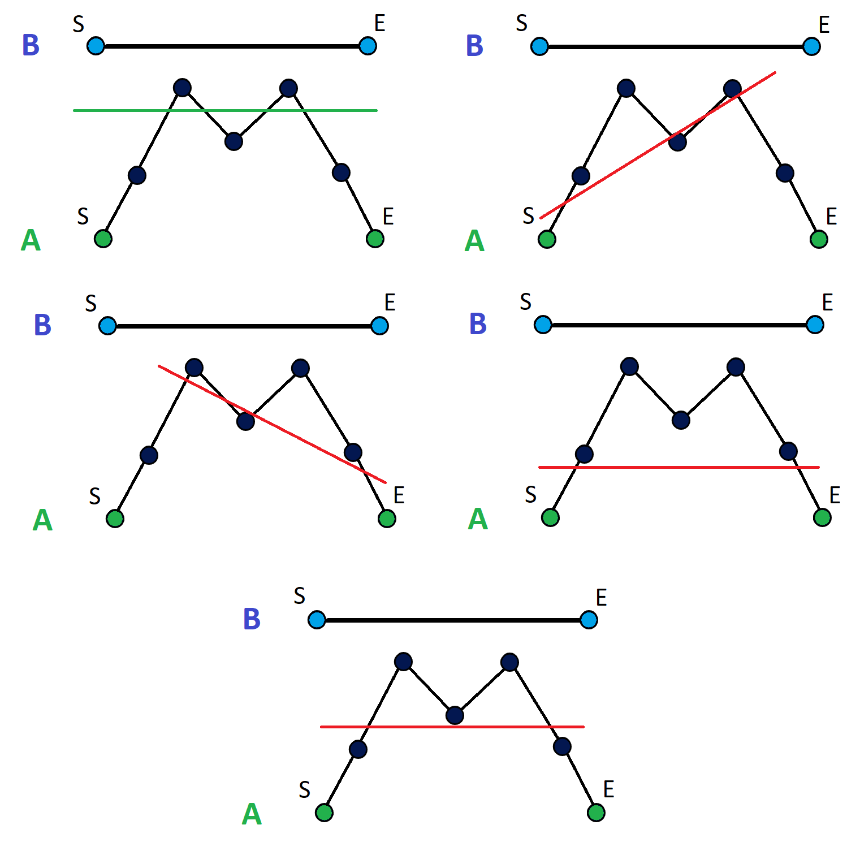
As one of the conditions is that these middle points must be between A and B, if two consecutive points from the convex hull do not reduce the total distance when moved into B, moving any of segment's inner points will not reduce the total distance either. If said inner points were moved, it would cause intersections between both paths, which is already an indication that it is not the optimal placement.



The rule of derivation consists on finding the appropiate reduction of the total distance between two MCs using the convex hull. The effect of moving the start and exit points of the closest segment to B is checked, then the first segment together with another of its consecutive ones, then the first one together with the second consecutive one, after that the first one together with both consecutive ones and so on successively. Finally, the inner points must be checked one by one, starting with those of the selected segments and ending with those of the non-selected consecutive segments.



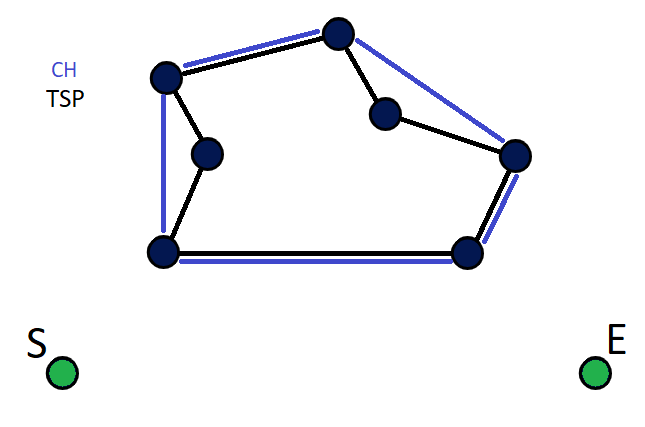
If there are any sets which minimizes the total distance, the largest one will be picked (in this case, the first set marked in green would be picked since the ones in red do not improve the total distance). The meaning behind these steps is to 'make methodic cuts' to A, knowing that there cannot be intersections. Another possible representation is as follows:



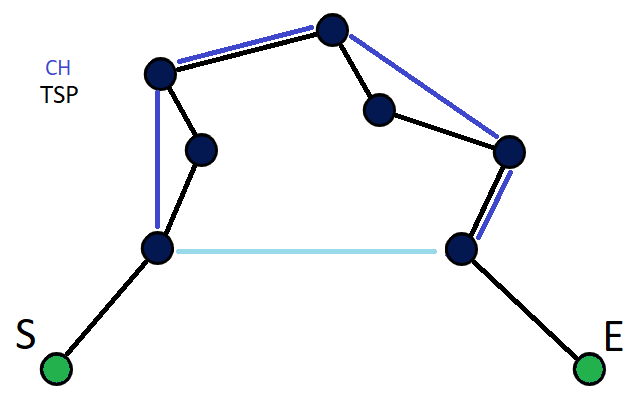
Once the set is detected, to move it into B their current order must be taken into account since these will follow the same order since the set can be either a segment of the MP (by the composition rule we know it will be another MP) or it will be part of the convex hull, which we know that it must be respected to avoid intersections.

## TSP to Minimum Path - REWORK (no llego a convertir un TSP en CM ya, lo único que debo de explicar es que un CM es como un TSP en el que se ha ignorado un lado (lado comprendido entre E y S)

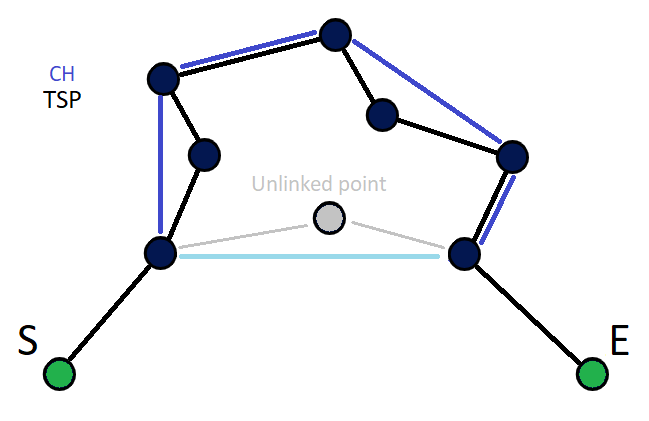
Given a TSP tour and a starting and exit point, the objective is to create a minimum path. There exists one restriction to the problem, which is that the line that links the start and the exit do not intersect with the TSP.



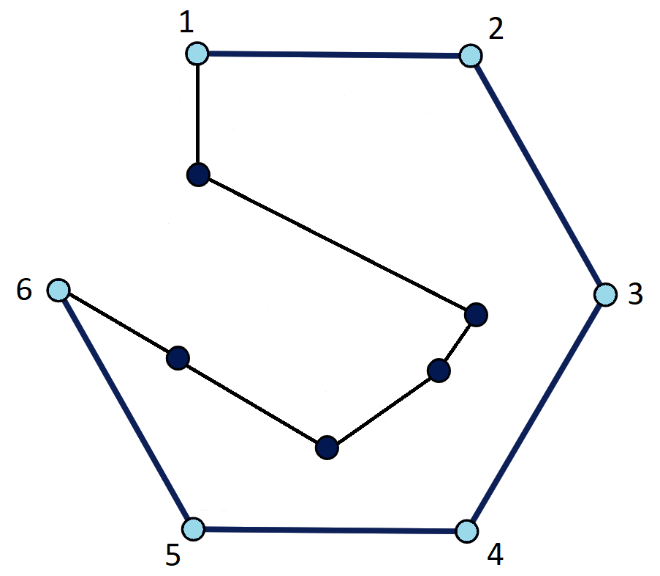
The main idea is to locate two points from the TSP to unlink one of their edges and connect themselves with the start and exit points. As it was previously stated, a TSP is a type of minimum path, and by the composition rule, it is possible to take any inner segment from it keeping its property of being of minimum length. Following this idea, if we take away an empty segment from the TSP's convex hull and use the rest of the path to connect it to the S/E points, the result will be a minimum path as well.



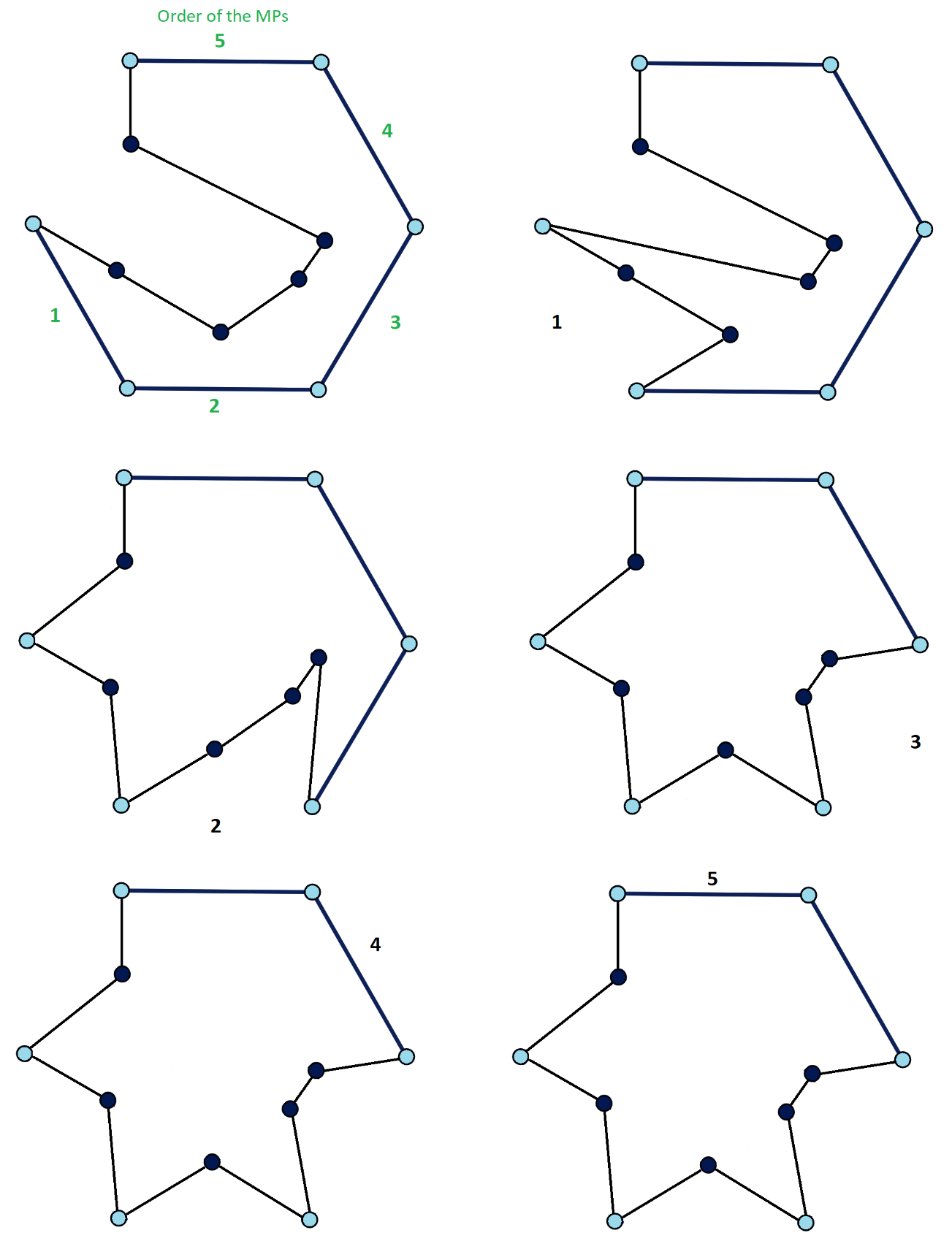
However, if there exists inner points in the segment facing S/E points, this process would leave those points unlinked:



To link these points into the path, we can consider this an instance of the TSP problem, with the spcial condition that the side between S and E is not considered when calculating it. Due to the constraints of this problem, we are certain that S/E points will always stay in the convex hull of the set of points, which is easy to identify.



By using the derivation rule explained previously, it is possible to optimize the total distance between the segment that holds the inner points and any of the other minimum paths in the TSP. A way of doing so is starting from one of the minimum paths adjacent to the 'holder' and applying the derivation rule. Then, following that order, the third minimum path would be optimized with both the holder and the previous minimum path, and so on until all MPs have been optimized.



After the procedure, all minimum paths will only contain the inner points required to minimize the total length, therefore reaching the solution of the TSP.

## Optimization technique for TSP

Starting from a convex hull and a minimum path which connects all inner points with two consecutive points of the convex hull, the goal is to optimize the TSP solution by moving the inner points to their corresponding minimum paths where they produce the least increment in distance (this of course follows the partition aforementioned).

## Designed algorithm: PolyGenesis - Necesidad de recursividad, puedo usar parte del código de Matlab como pseudocódigo.

Concepto de topologia (sigue siendo el mismo casi) y ya el desarrollo. Lo dejo pendiente para cuando tenga el código por si quiero meter pseudo o lo que sea.

# Results

# Conclusions and future work