Concurso Modelización Matemática
Problema de Steiner

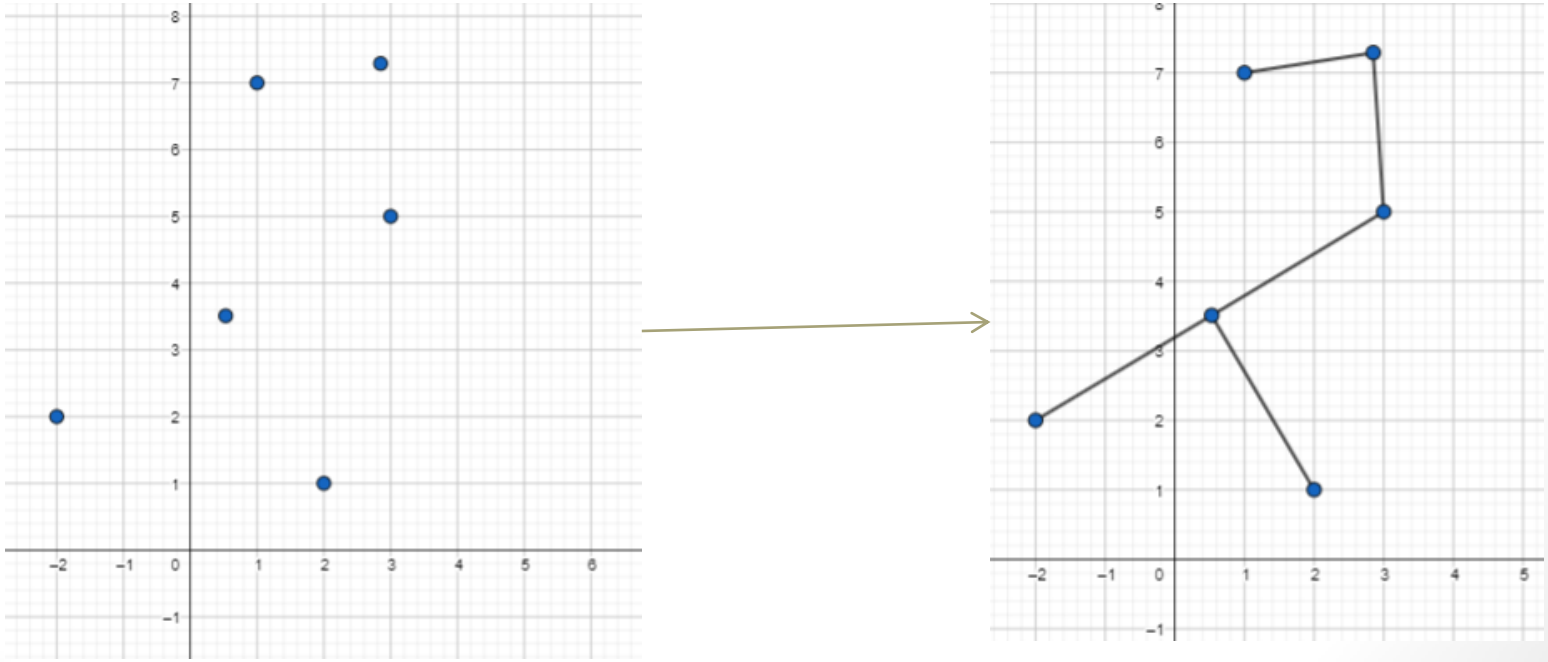
Problema dividido en 2 partes:

1. Problema de árbol mínimo euclídeo.
2. Problema de árbol de Steiner.

Problema de árbol mínimo euclídeo

Problema: Dado un conjunto de puntos en el plano, encontrar la forma de conectarlos todos entre sí tal que la longitud total de todos los “cables” sea la menor posible.

Es un problema conocido, llamado problema del árbol mínimo euclídeo. Estudiado desde hace muchos años y con soluciones muy conocidas y muchas aplicaciones.



Propiedades de una solución

Se puede probar que una solución al problema cumple:

1. La red no tiene ciclos.

(árbol)

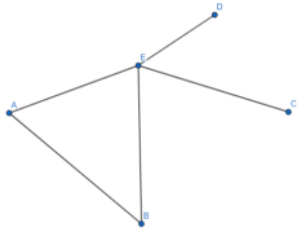


Figura 1: Red con un ciclo

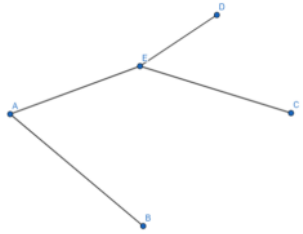
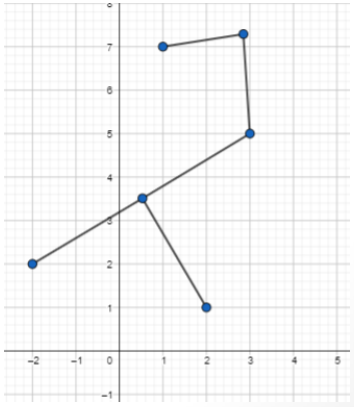


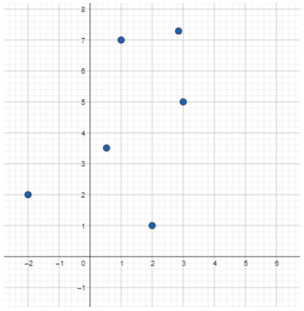
Figura 2: Red sin ciclos

2. Entre dos puntos de la red hay un

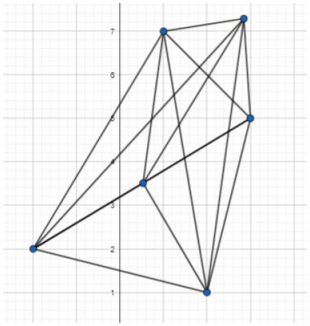


Algoritmo para encontrar dicho grafo

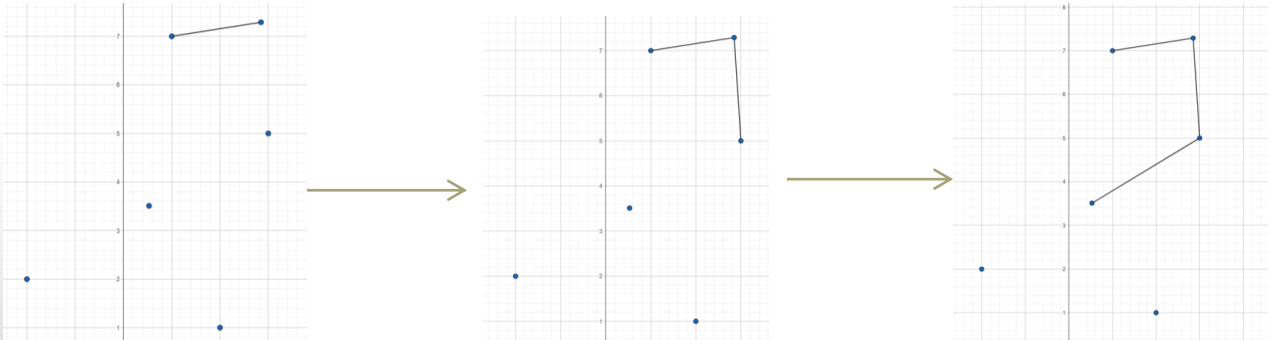
Estudiado desde hace muchos años y con soluciones muy conocidas y muchas aplicaciones. Uno de los algoritmos es el de Prim:



1. Considerar todas las conexiones entre los puntos.



2. Ir agregando la línea más corta posible tal que no se generen ciclos hasta conectar todo el grafo



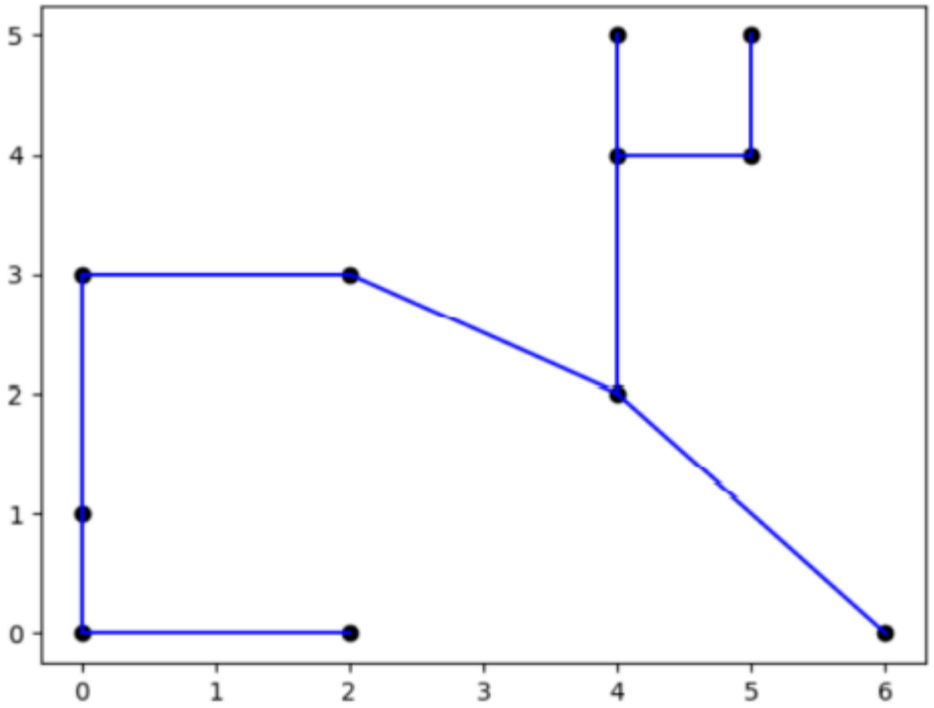
Se puede probar que el algoritmo de Prim llega a la solución óptima.

Tiene una complejidad temporal de $O(V^2 \log V)$

Con V el número de vértices. Hay otros algoritmos para este mismo problema como el de Kruskal con la misma complejidad temporal.

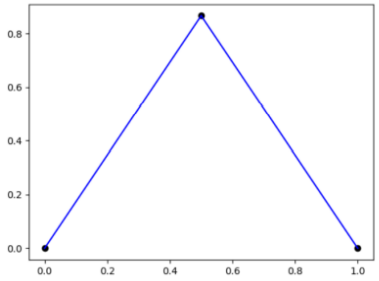
Se puede mejorar la complejidad a O(V log V) con un método no considera todas las conexiones entre puntos al inicio.

Se implementó el algoritmo en Python, para dados los puntos en el plano, construir el árbol recubridor mínimo.

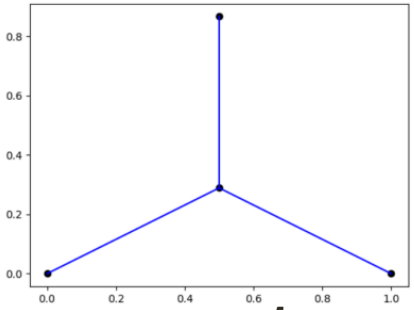


Mejorar el grafo: Problema de Steiner

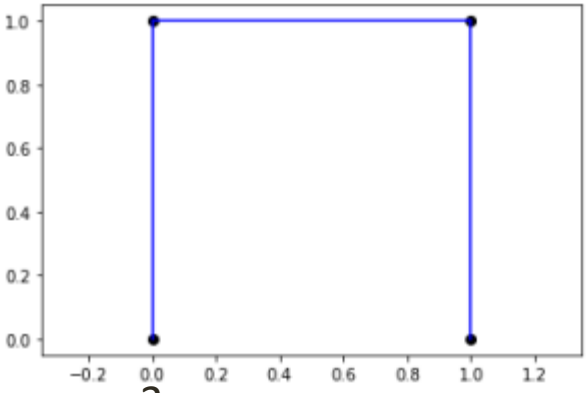
En ciertas ocasiones, agregar puntos al grafo puede disminuir el tamaño de la red completa.



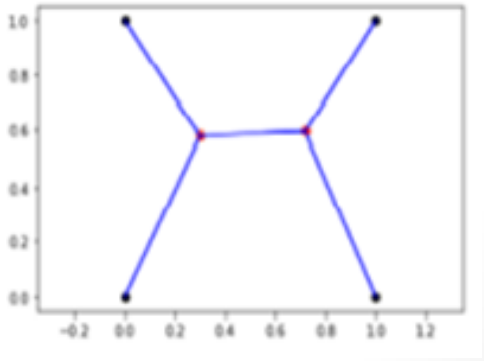
2 cm



1.73 cm = $\sqrt{3}$ cm



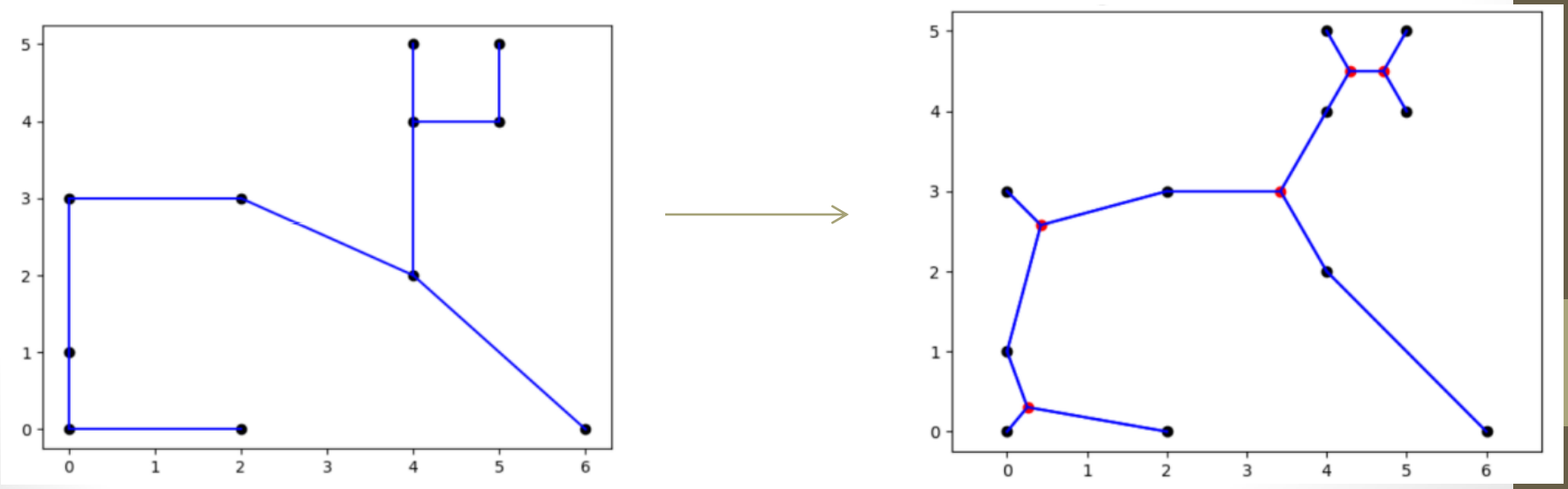
3 cm



2.732 cm

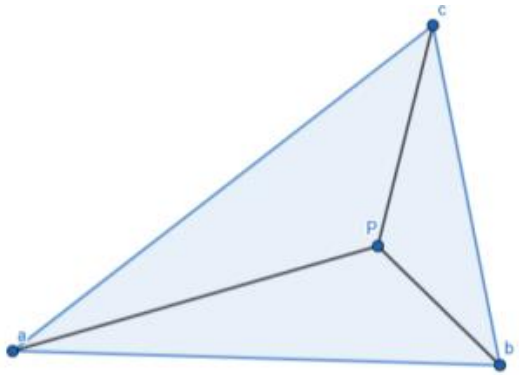
Problema del árbol de Steiner Euclídeo:

Dado un conjunto de puntos en el plano, encontrar la forma de conectarlos todos entre sí tal que la longitud total sea la menor posible, con la posibilidad de agregar puntos adicionales (puntos de Steiner).



Problema particular: Problema de Fermat

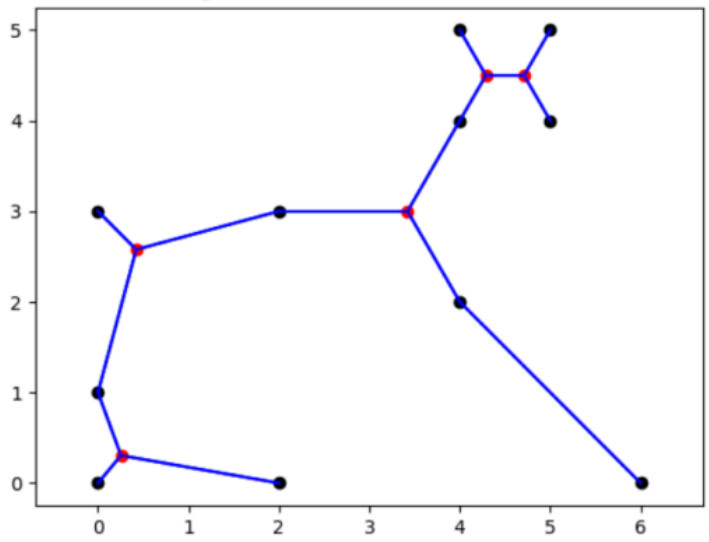
Dado un triángulo, encontrar el punto tal que la distancia total desde los vértices del triángulo hasta el punto sea la menor posible. Es un caso particular del problema de Steiner para 3 puntos.



El problema tiene una solución que se puede construir geométricamente. Se puede probar que el punto P es distinto a los vértices del triángulo si y sólo si todos los ángulos del triángulo son menores a 120 grados. Y se puede probar que los ángulos de las 3 aristas que se unen en P son siempre de 120 grados.

Propiedades

1. Todos los puntos de Steiner tienen exactamente 3 aristas, que se juntan en ángulos de 120 grados.
2. Si la cantidad de vértices originales es V, a lo sumo habrá que agregar V-2 puntos de Steiner para llegar a la solución óptima.
3. La longitud del árbol con puntos de Steiner es en el mejor de los casos un $\sqrt{3}/2$ % = 86.6 % la longitud sin haberlos agregado.



Algoritmo para encontrar el árbol de Steiner

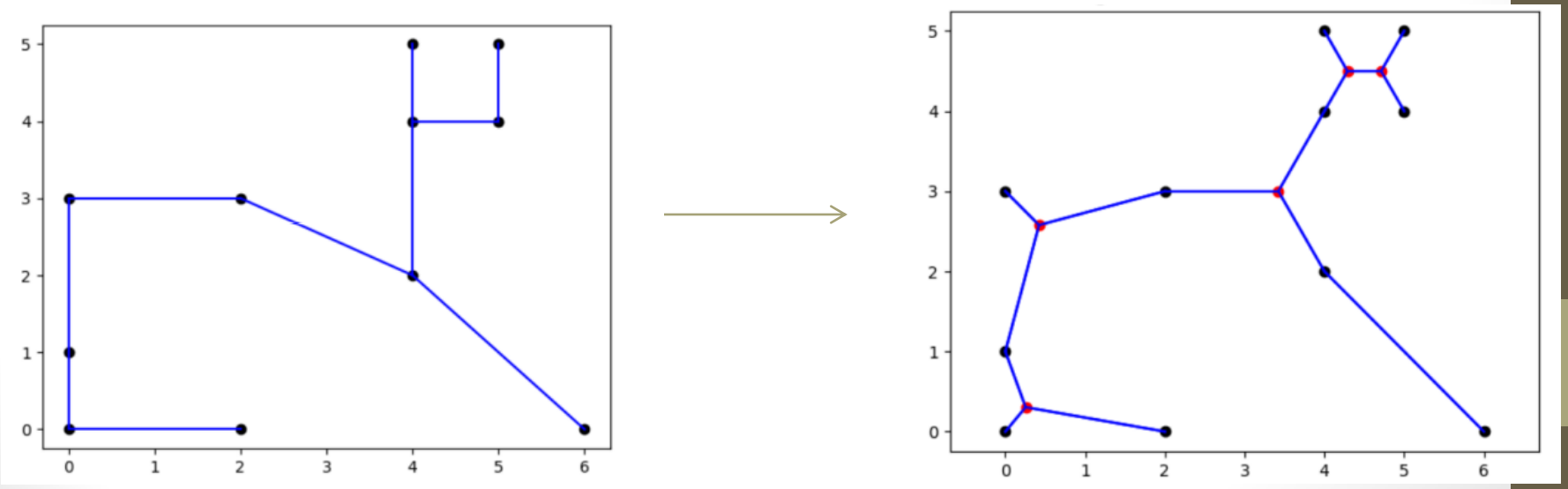
El problema de Steiner es mucho más difícil de resolver, pues es difícil decidir cuántos puntos vamos a agregar al grafo y dónde los vamos a colocar, etc.

Es un problema NP-duro, lo que significa que no se puede resolver en un tiempo polinomial. Entonces sólo se pueden encontrar soluciones muy lentas pero óptimas o que no sean óptimas pero quizá sean rápidas.

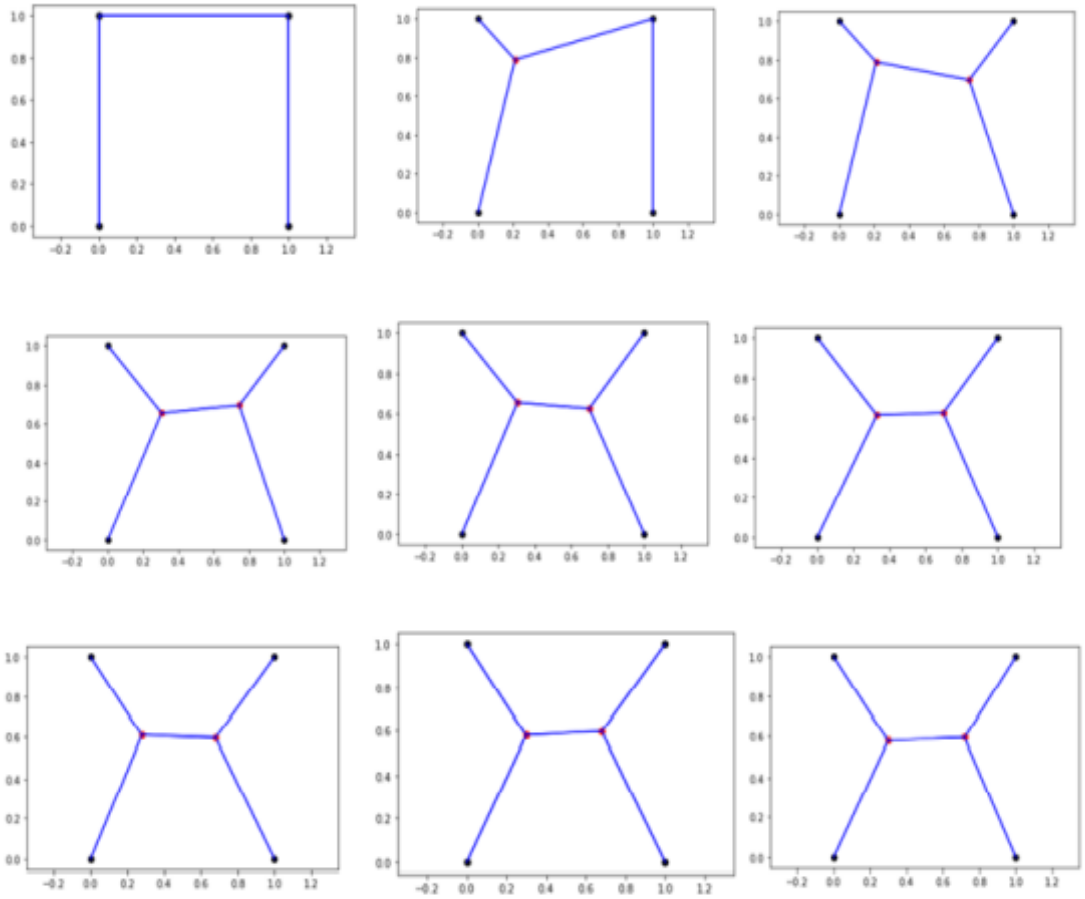
Además, es un problema mucho menos estudiado y existe muy poca literatura al respecto.

Algoritmo propuesto

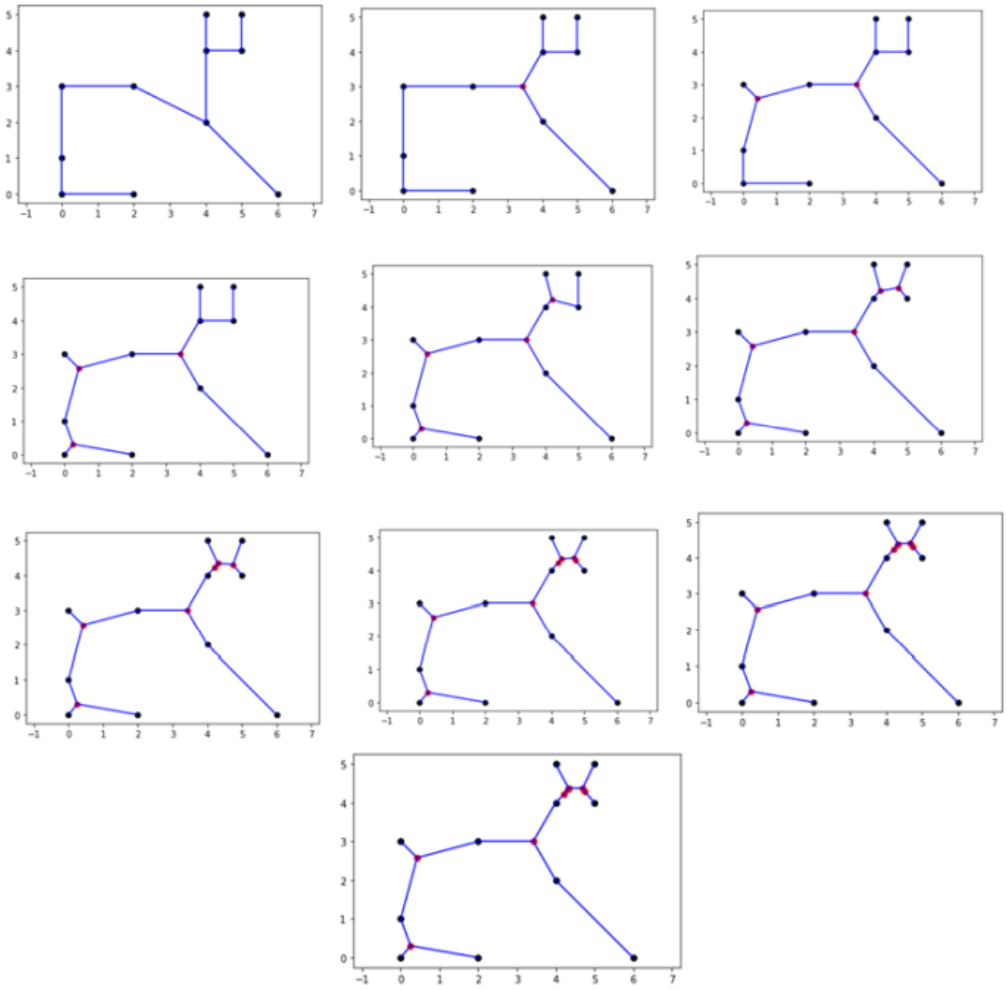
1. Encontrar el árbol recubridor mínimo del conjunto de puntos originales.
2. Ir recorriendo cada trío de puntos conectados en el árbol, agregar el punto de Fermat de dicho triángulo y rehacer las conexiones.
3. Repetir hasta que ya no se puedan agregar puntos de Fermat en ningún triángulo.



Ejemplo 1

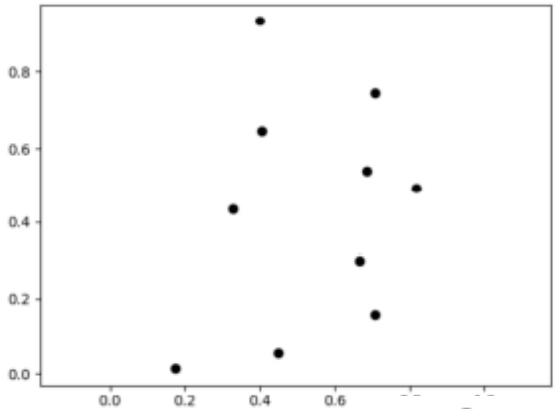


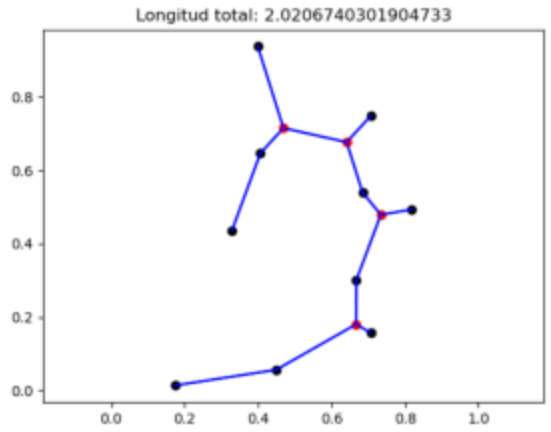
Ejemplo 2

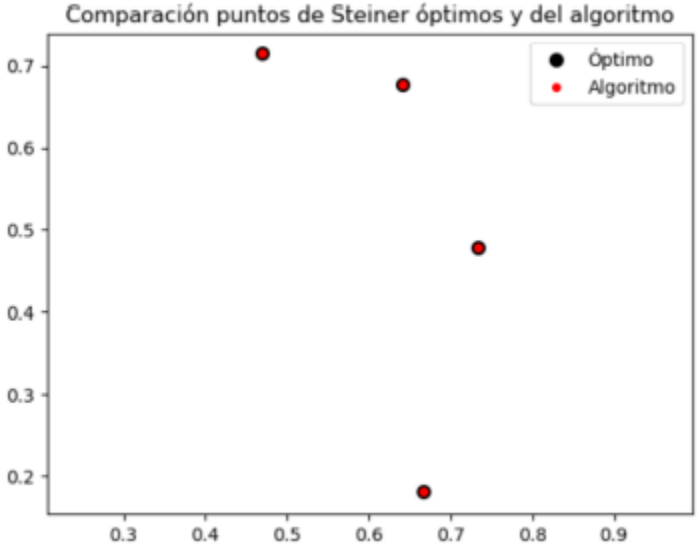


El algoritmo propuesto no llega siempre a la solución óptima pero es rápido y se acerca bastante a la solución óptima.

Poner a prueba el algoritmo

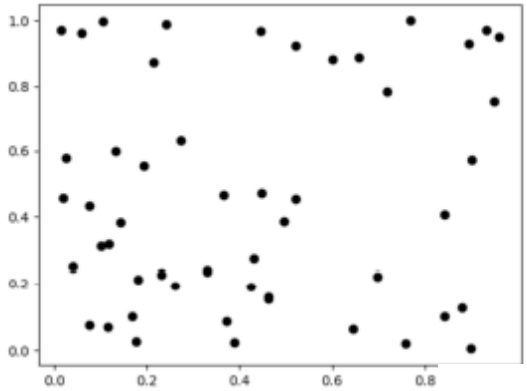


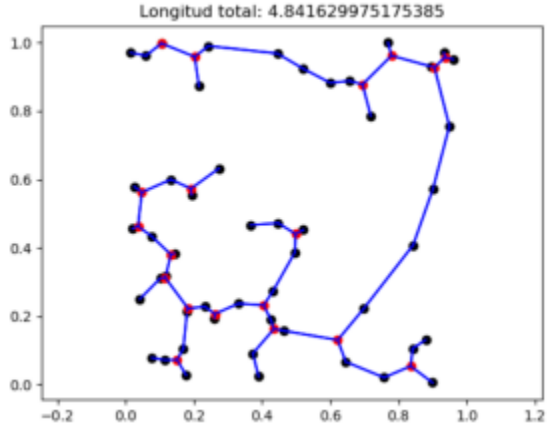


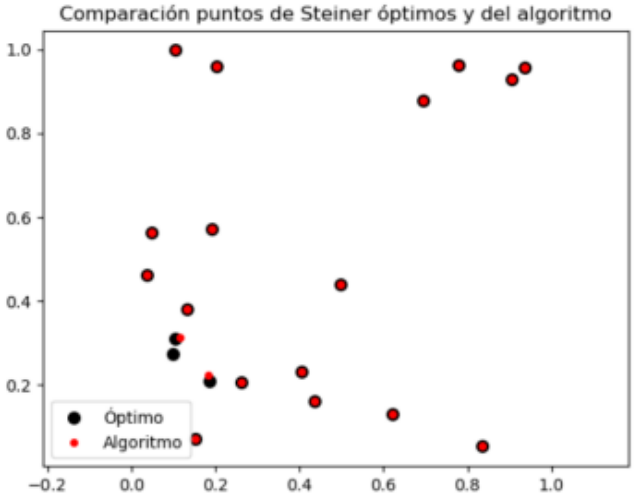


Longitud óptima = 2,020673795 Longitud según algoritmo = 2,020673814

Error porcentual = $9{,}02 \times 10^{-7}\,\%$

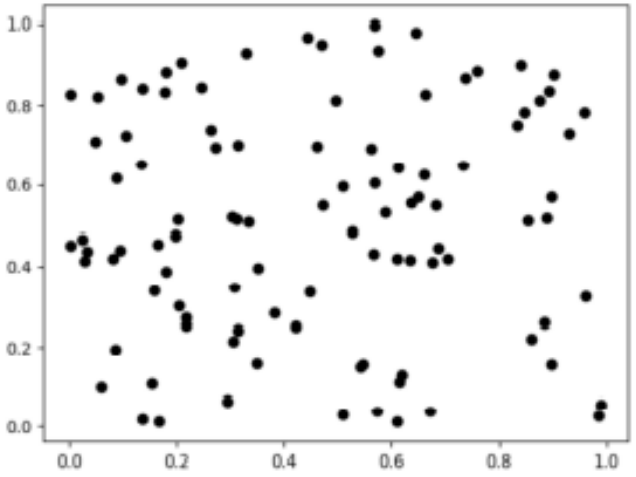


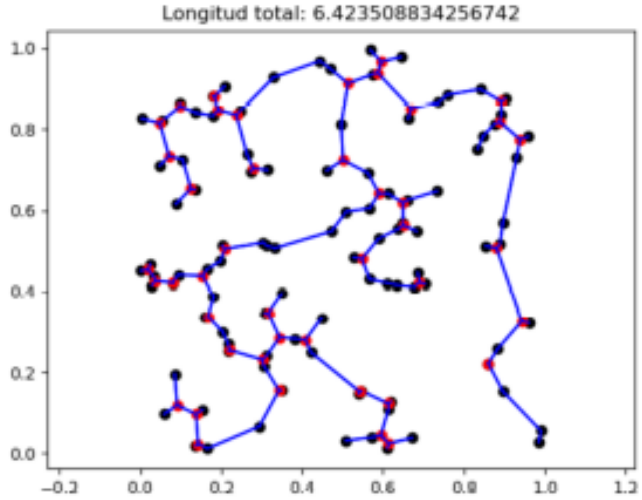


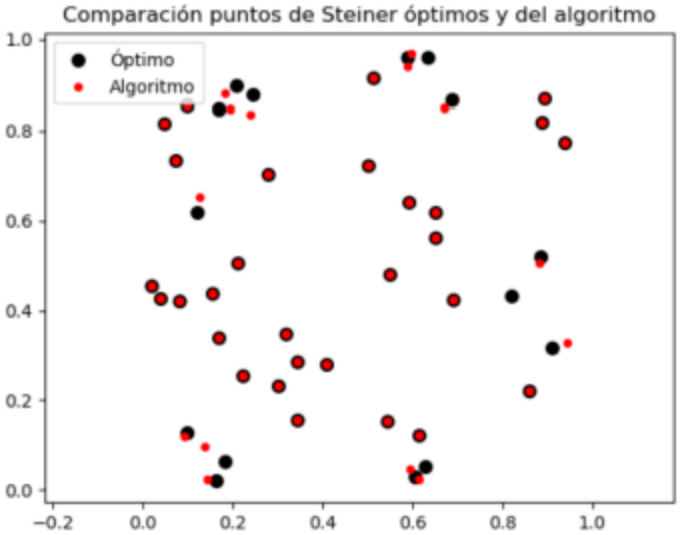


Longitud óptima = 4,836601423 Longitud según algoritmo = 4,841629975

Error porcentual = 0,1039687 %







Longitud óptima = 6,394256 Longitud según algoritmo = 6,423508834

Error porcentual = 0,457486125 %

	10 Nodos		30 Nodos		50 Nodos		70 Nodos		100 Nodos	
	Tiempo [s]	Error porcentual [%]	Tiempo [s]	Error porcentual [%]	Tiempo [s]	Error porcentual [%]	Tiempo [s]	Error porcentual [%]	Tiempo [s]	Error porcentual [%]
Problema 1	0.00701	9.40E-07	0.0309	0.092	0.3481	0.1039	0.769	0.0335	1.669	0.4574
Problema 2	0.004982	0	0.08477	0.0369	0.375	0.9327	0.3261	1.4286	1.665	0.6119
Problema 3	0.006014	2.65E-06	0.04639	0.3856	0.1864	0	0.6931	0.3591	1.392	0.4004
Problema 4	0.002991	0	0.1675	1.91E-06	0.4059	0	0.6352	0.178	1.551	0.507
Problema 5	0.003955	0.00	0.0588	0.2329	0.3929	0.023	0.653	0.5601	1.3561	0.1635
Problema 6	0.005017	5.06E-06	0.1027	0.1898	0.375	0.612	0.806	0.242	1.2043	0.4722
Problema 7	0.003988	5.98E-03	0.1366	0	0.4209	0.2065	0.673	0.1009	1.2301	0.77651
Problema 8	0.017985	0.00	0.1327	0.1103	0.2293	0.4789	0.701	0.772	1.1951	0.4135
Promedio:	0.00649275	0.00074858	0.095045	0.13093774	0.3416875	0.294625	0.65705	0.459275	1.407825	0.47530125

Tabla de resultados