Tarea 2: Recocido simulado

Fernando Carrillo A01194204

3/09/20

Descripción del problema:

Se va a intentar resolver el problema del vendedor viajero utilizando el método de recocido simulado. El problema del vendedor viajero consiste en tener una cantidad de ciudades que se desean recorrer en un mapa, y encontrar la ruta óptima que pase por todas las ciudades una vez y regrese al inicio en la menor distancia posible.

Descripción de la representación de los individuos:

Para este problema se van a representar las ciudades como puntos/coordenadas en un plano cartesiano. Estas ciudades estarán dentro de un rango de [1:50] en ‘x’ y ‘y’.
Las distancias recorridas entre dos ciudades estarán representadas como distancias euclidianas. No se especificó métrica de distancia, por lo cual se manejarán como unidades de distancia.

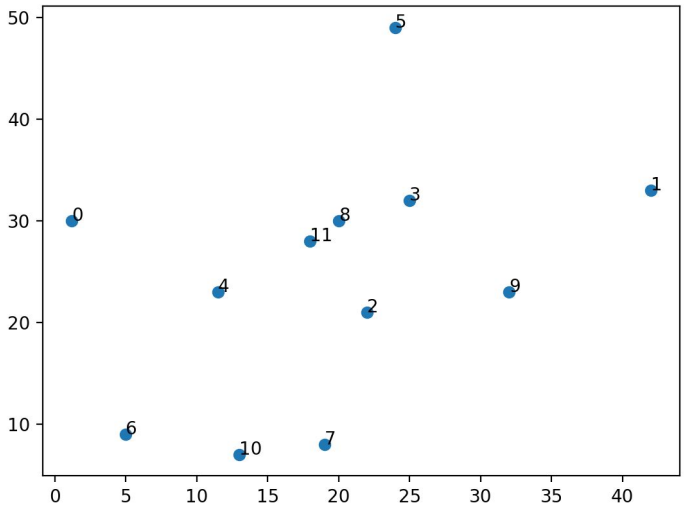
Descripción de la función de evaluación:

La función de evaluación toma una ruta con las ciudades a recorrer, y calcula la distancia total que tomaría recorrer todas las ciudades en el orden especificado.

Resultados obtenidos:

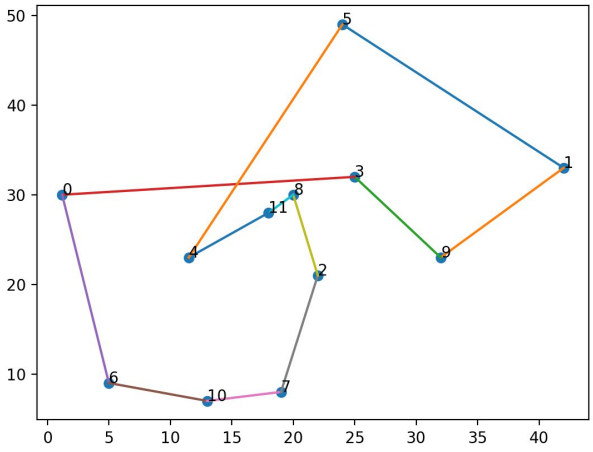
Al principio las coordenadas de las ciudades estaban siendo creadas aleatoriamente con cada ejecución del programa, pero con fines de probar cambios en los parámetros y buscar mejoras, se creó una matriz con doce ciudades fijas.

(Plano cartesiano con ciudades utilizadas para realizar las pruebas)

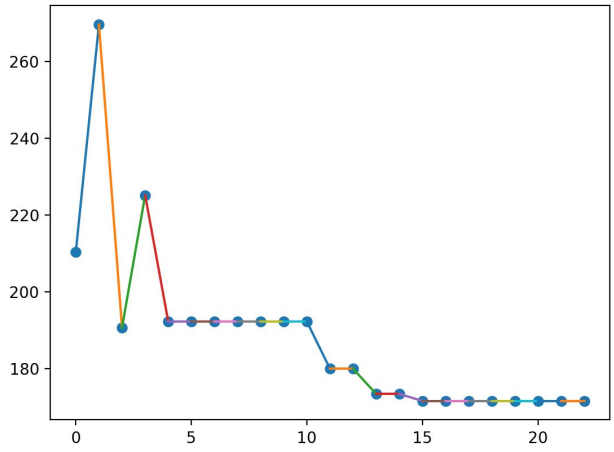


Después de terminar la primera iteración del algoritmo, se obtuvieron los siguientes resultados en diferentes ejecuciones del programa:

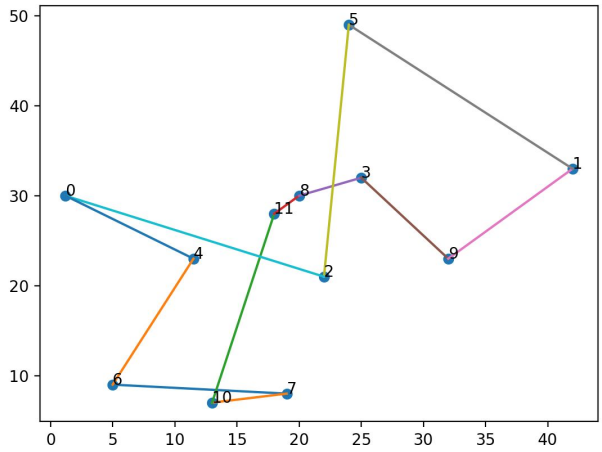
Ruta 1



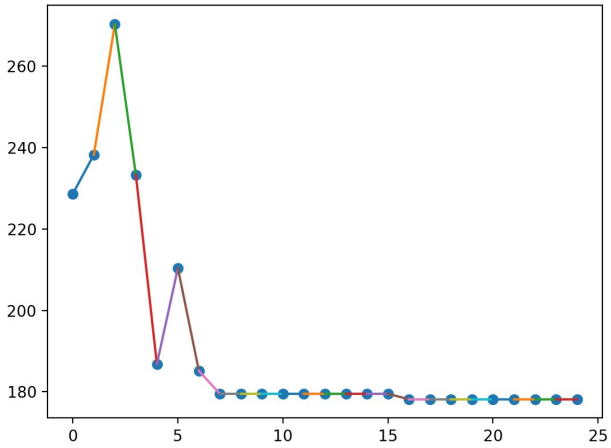
Mejor distancia 171.62 unidades



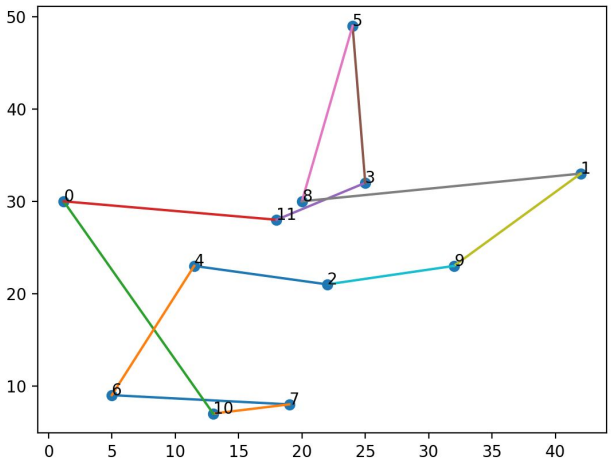
Ruta 2



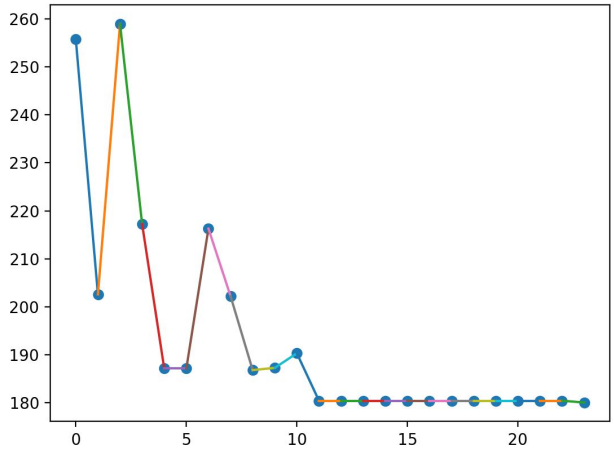
Mejor distancia 178.17 unidades



Ruta 3



Mejor distancia 180.06 unidades



Como se puede observar en las gráficas, el algoritmo demuestra una mejora en la distancia total conforme pasan las iteraciones, pero no es muy consistente. También podemos ver el proceso de selección de una solución peor cuando la temperatura es alta al inicio, y como se va estabilizando conforme la temperatura baja en cada iteración.

Creo que es importante tomar nota sobre los siguientes puntos sobre la implementación de esta primera iteración del algoritmo:

- La función para generar un vecino es completamente estocástica. En cada ejecución recibe una ruta y hace un ‘shuffle’ de las ciudades
- El parámetro L que limita la longitud de las cadenas de Markov, y que limita las iteraciones en cada cambio de temperatura está limitado a 100
- Los valores de Alpha (0.7) y Beta (1.5)

- El algoritmo se detiene cuando la temperatura baja sobre 0.01, o se obtienen los resultados más de 15 veces seguidas
- Para la inicialización de la temperatura, se busca que se alcance a un porcentaje de aceptación de mínimo r_min = 0.6

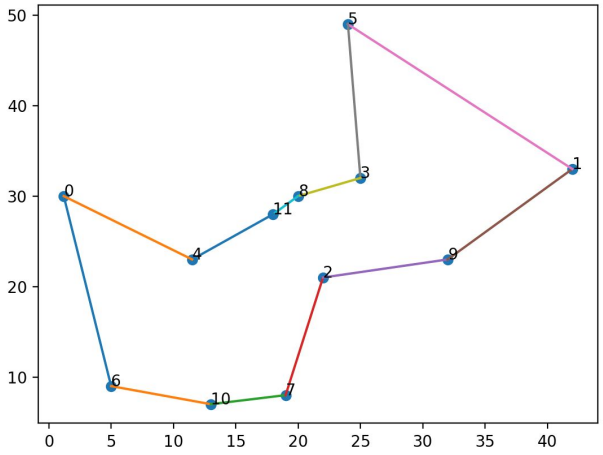
Para mejorar los resultados del algoritmo, se hizo un cambio a la función que genera vecinos. Ahora en lugar de simplemente hacer un 'shuffle' de las ciudades en la ruta, se hacen swaps de las ciudades consecuentes en la ruta. La decisión de hacer o no un swap varía dependiendo de una probabilidad que incrementa conforme la temperatura.

También se actualizaron los siguiente parámetros:

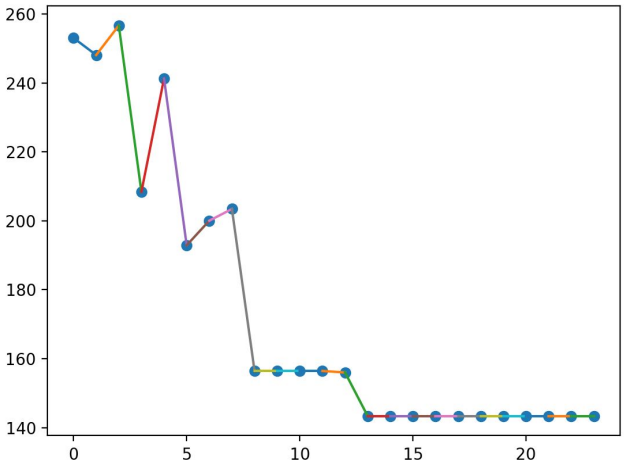
- Se elevó r_min = 0.7, lo cual implica una mayor temperatura inicial

Resultados de cambiar la función que genera un vecino:

Ruta 4

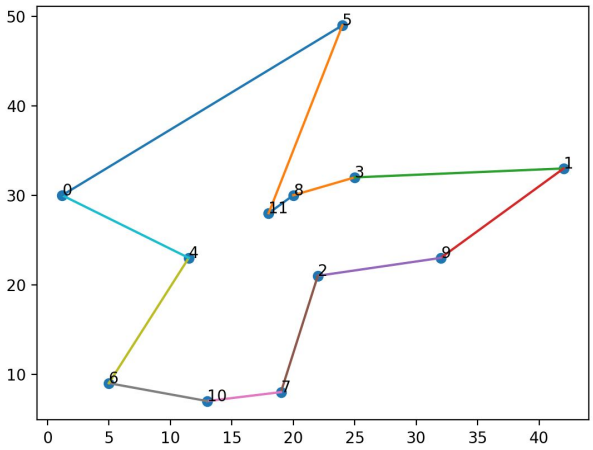


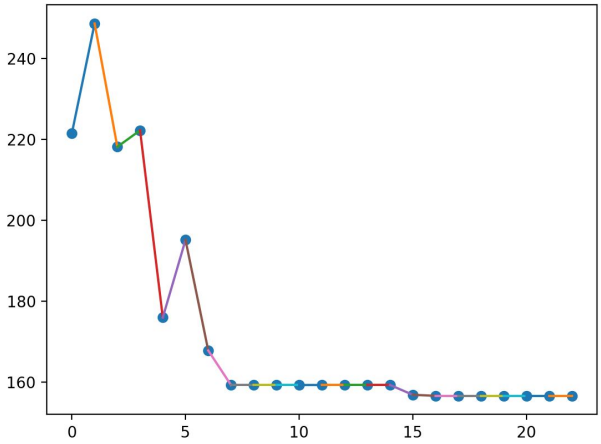
Mejor distancia 143.33 unidades



Ruta 5

Mejor distancia 153.50 unidades



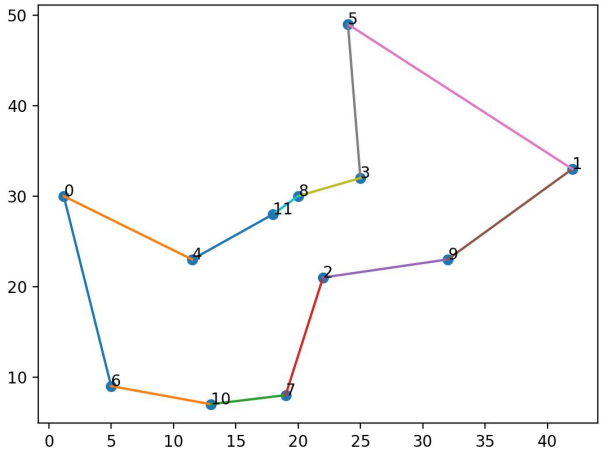


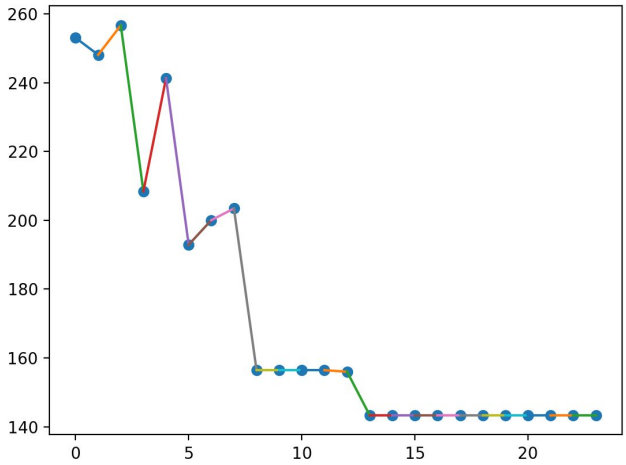
Los resultados mejoraron considerablemente. Aunque las diferentes ejecuciones daban diferentes resultados, las distancias bajaron de un rango de 170-180 unidades a 140-160 unidades de distancia.
De igual manera parece que la Ruta 4 fue la mejor solución, se repitió en varias ocasiones y no hubo alguna con menor distancia.

Gráfica con la curva de mejor encontrado:

Ruta 4

Mejor distancia 143.33 unidades





Esta ruta fue la que obtuvo la menor distancia, se llegó a esta solución en varias ejecuciones del programa después de hacer los cambios a la función que genera el vecino.

Conclusiones y retos encontrados:

Esta tarea me ayudó bastante a entender el algoritmo, puesto que lo tuve que descomponer en sus diferentes subprocesos para luego integrarlo. Al principio tuve dificultad entendiendo algunos parámetros como ‘L’, y cómo era el proceso de las cadenas de Markov, pero luego entendí que era parte del proceso del incremento de temperatura, para después empezar el enfriamiento.

Yo creo que el algoritmo es muy útil, aunque algo inconsistente cuando lo corres una vez, pues no siempre encontrará el mínimo global. Lo que buscaría hacer es correr un ‘batch’ de simulaciones y escoger la mejor solución de todas. De esta manera buscaría ampliar la búsqueda del mínimo global en caso de que algunas de las simulaciones no logren salir de sus mínimos locales por cuestiones de la naturaleza estocástica.

De igual manera podría mejorar el algoritmo que genera vecinos, y jugar con los parámetros de Alpha, Beta, L, y las condiciones de paro para mejorar su ejecución.