**1. Explicación del Problema del Viajante**

El **Problema del Viajante** o **TSP (Travelling Salesman Problem)** es un problema clásico de optimización que plantea la siguiente cuestión:

Dada una lista de ciudades y las distancias entre cada par de ellas, ¿cuál es la ruta más corta posible que visita cada ciudad exactamente una vez y regresa a la ciudad de origen?

Este problema es fundamental en optimización combinatoria y pertenece a la clase de problemas **NP-difíciles**, lo que significa que no existe una solución exacta en tiempo polinómico conocida. Esto es especialmente relevante porque, a medida que se incrementa el número de ciudades, las combinaciones posibles de rutas crecen factorialmente. El problema tiene numerosas aplicaciones, desde la logística y planificación de rutas hasta la fabricación de circuitos electrónicos y secuenciación de ADN​(Problema del viajante).

**2. Cálculo de las combinaciones posibles para 15 ciudades**

Para calcular el número de combinaciones posibles de rutas en el TSP con **15 ciudades**, se utiliza la fórmula de permutaciones ajustada para considerar rutas únicas de ida y vuelta:

donde NNN es el número de ciudades.

En este caso:

Calculamos esto paso a paso:

* 14! (factorial de 14) equivale a 87.178.291.200 posibles combinaciones.
* Dividiendo por 2, obtenemos:

Así, para 15 ciudades, existen **43.589.145.600 rutas posibles**.

**3. Resumen de Métodos Heurísticos para Resolver el TSP**

Dado que el TSP es NP-difícil, los algoritmos exactos no son prácticos para grandes instancias. Por eso se emplean métodos **heurísticos**, que proporcionan soluciones cercanas al óptimo en menor tiempo. Algunos de los métodos heurísticos más destacados incluyen:

* **Algoritmo del Vecino Más Cercano (Nearest Neighbor)**: Este es un método voraz en el cual el viajante selecciona en cada paso la ciudad no visitada más cercana. Aunque es rápido, no garantiza la solución óptima y puede dar lugar a rutas subóptimas si las distancias entre ciudades están desbalanceadas.
* **Algoritmo de 2-Opt**: Este método busca mejorar una ruta inicial eliminando pares de aristas y reconectando los fragmentos de manera que se reduzca la distancia total. Es un caso específico del método **k-opt** que permite mejorar iterativamente una solución inicial.
* **Algoritmo de Christofides**: Este algoritmo garantiza una solución que no es más del 50% superior al óptimo en el caso de TSP con distancias métricas (aquellas que cumplen la desigualdad triangular). Combina árboles de expansión mínima y emparejamientos para crear rutas eficientes.
* **Recocido Simulado (Simulated Annealing)**: Inspirado en la metalurgia, este algoritmo ajusta progresivamente las soluciones, permitiendo ocasionalmente soluciones peores para escapar de mínimos locales, con el objetivo de acercarse al óptimo a lo largo del tiempo.
* **Algoritmo de Colonia de Hormigas (Ant Colony Optimization)**: Basado en el comportamiento de las hormigas, este algoritmo simula una colonia en la cual los agentes (hormigas) exploran distintas rutas y "marcan" las buenas soluciones con feromonas, lo que guía a otras hormigas hacia rutas favorables.