

# Actividad Práctica: Problema del Viajante sobre Grafos Completos

Curso de Teoría de Grafos

## Información general

**Modalidad:** Grupos de máximo 3 personas.

### 1. Planteamiento del problema

Un viajante necesita visitar un conjunto de ciudades exactamente una vez cada una y regresar a la ciudad de origen, minimizando la distancia total recorrida. Este problema clásico, conocido como el Problema del Viajante (TSP, por sus siglas en inglés), es uno de los más estudiados en teoría de grafos y optimización combinatoria debido a su dificultad computacional y sus múltiples aplicaciones prácticas en logística, planificación de rutas y diseño de redes.

La pregunta fundamental es: *¿Cuál es la mejor ruta posible?* Para instancias pequeñas es posible examinar todas las rutas posibles y garantizar que se encuentra el óptimo absoluto, pero esta estrategia se vuelve rápidamente intratable cuando el número de ciudades aumenta debido al crecimiento factorial del espacio de soluciones. Por otro lado, existen métodos heurísticos que construyen soluciones rápidamente siguiendo reglas simples, sin garantizar optimalidad pero ofreciendo tiempos de ejecución razonables incluso para problemas grandes.

### 2. El desafío

Su tarea consiste en resolver una instancia real del TSP y experimentar directamente el dilema fundamental entre *certeza y velocidad*: métodos exactos que garantizan el óptimo versus heurísticas rápidas que producen soluciones aproximadas.

**Contexto de la instancia:** Seleccionen entre 6 y 9 ciudades reales de su interés (pueden ser ciudades de Chile, de su región, de Sudamérica, o de cualquier parte del mundo). Obtengan las coordenadas geográficas (latitud, longitud) de cada ciudad. Con esta información, construyan un grafo completo  $K_n$  donde cada vértice representa una ciudad y cada arista tiene un peso correspondiente a la distancia euclidiana entre las coordenadas. Calculen la matriz de distancias  $D = [d_{ij}]$  que representa todas las distancias entre pares de ciudades.

**Primera aproximación – búsqueda exhaustiva:** Implementen un algoritmo de búsqueda exhaustiva que examine sistemáticamente todos los ciclos Hamiltonianos posibles. Para cada ciclo candidato, calculen su longitud total (suma de distancias de las aristas recorridas). Al finalizar la enumeración completa, identifiquen el ciclo óptimo  $\pi^*$  con longitud mínima  $L^*$ . Este método les garantiza encontrar la mejor solución posible, pero ¿cuánto tiempo toma? ¿Qué tan grande puede ser  $n$  antes de que el cálculo se vuelva impracticable?

El código que desarrollen debe no solo calcular la solución óptima, sino también representarla visualmente. Generen una animación o secuencia de figuras que muestre el proceso de búsqueda: cómo se evalúan diferentes ciclos, cuál es el mejor encontrado hasta el momento, y cómo finalmente se converge al óptimo global. Esta visualización debe permitir entender *cómo* funciona el algoritmo exhaustivo.

**Segunda aproximación – heurística constructiva:** Implementen ahora la heurística de Vecino Más Cercano (Nearest Neighbor). Comenzando desde una ciudad inicial, construyan el ciclo paso a paso: en cada iteración, desde la ciudad actual, viajen a la ciudad no visitada más cercana. Continúen hasta visitar todas las ciudades y regresar al origen. Obtengan la solución heurística  $\pi^{\text{NN}}$  con longitud  $L^{\text{NN}}$ . Esta estrategia es mucho más rápida que la búsqueda exhaustiva, pero ¿qué tan buena es la solución encontrada?

Al igual que en el caso anterior, su código debe generar una visualización del proceso: una secuencia de figuras que muestre cómo se construye el ciclo paso a paso, qué ciudad se selecciona en cada iteración, y por qué (es la más cercana disponible). Esta visualización debe revelar la naturaleza *greedy* (miope) del algoritmo.

**Comparación y análisis:** Una vez implementados ambos métodos, compárenlos cuantitativamente. Calculen el *gap de optimalidad*:  $g = \frac{L^{\text{NN}} - L^{\star}}{L^{\star}} \times 100\%$ . Este valor indica qué porcentaje más larga es la solución heurística respecto al óptimo. Midan también los tiempos de ejecución de ambos algoritmos. ¿Cuánto más rápido es el método heurístico? ¿Vale la pena el tiempo adicional invertido en búsqueda exhaustiva para su instancia particular?

Si observan que ambos métodos producen resultados muy similares (gap pequeño), aumenten gradualmente el número de ciudades hasta que comience a aparecer una diferencia significativa. Esto les permitirá entender mejor en qué tipo de instancias las heurísticas pueden fallar.

### 3. Formato de entrega

#### Fechas

**Informe escrito:** Sábado 22 de noviembre hasta las 23:59 hrs (formato PDF).

**Presentación oral:** Viernes 28 de noviembre en horario regular de clases.

#### Informe técnico

Elaborar un documento en formato PDF de 6–8 páginas que incluya:

- **Introducción:** Contexto del Problema del Viajante, relevancia, y descripción de la instancia particular resuelta (número y nombres de las ciudades seleccionadas).
- **Datos:** Tabla con las coordenadas geográficas de cada ciudad (incluir fuente). Matriz de distancias  $D$  calculada. Figura mostrando la ubicación geográfica de las ciudades.
- **Metodología:** Descripción detallada de los dos algoritmos implementados (búsqueda exhaustiva y Vecino Más Cercano). Explicación del sistema de visualización desarrollado. Lenguaje de programación y librerías utilizadas.
- **Resultados:** Presentación de ambas soluciones:  $\pi^{\star}$  con longitud  $L^{\star}$  (óptimo) y  $\pi^{\text{NN}}$  con longitud  $L^{\text{NN}}$  (heurística). Figuras de ambos ciclos sobre las coordenadas geográficas. Capturas representativas o enlaces a las animaciones generadas. Tabla comparativa con longitudes, gap de optimalidad, y tiempos de ejecución.
- **Análisis:** Interpretación del gap obtenido. Comparación de tiempos de ejecución. Análisis de escalabilidad: estimación de tiempos para valores mayores de  $n$ . Discusión sobre el trade-off entre optimalidad y eficiencia observado. Limitaciones encontradas en las implementaciones.
- **Conclusiones:** Síntesis de los hallazgos principales. Reflexión sobre cuándo es apropiado usar cada método. Posibles extensiones del trabajo.
- **Referencias y código:** Bibliografía consultada. Fragmentos esenciales del código en el documento. Enlace a repositorio completo (GitHub, GitLab, o similar) con código fuente documentado, datos utilizados, y visualizaciones generadas.

## Presentación oral

Exposición de 10–15 minutos que debe cubrir:

- Presentación de la instancia del problema (ciudades seleccionadas y justificación).
- Explicación de los métodos implementados (conceptos clave de cada algoritmo).
- Demostración de las visualizaciones generadas (mostrar animaciones o secuencias).
- Reporte de resultados cuantitativos (rutas, longitudes, gap, tiempos).
- Análisis comparativo y conclusiones.

La presentación debe ser técnicamente rigurosa pero clara, permitiendo responder preguntas sobre decisiones de implementación y hallazgos.

## Rúbrica de evaluación

### Documentación (50 % de la nota)

Criterio	No presentado (0)	Deficiente (1)	Aceptable (2)	Completo (3)
<b>Formulación del problema</b>	Sin formulación	Errores conceptuales graves	Define el problema con imprecisiones menores	Formulación clara, precisa, con notación correcta
<b>Datos</b>	Sin coordenadas o matriz	Coordenadas sin fuente o matriz incorrecta	Coordenadas con fuente pero matriz incompleta	Coordenadas documentadas y matriz $D$ correcta
<b>Implementación exhaustiva</b>	Sin implementación	No funciona o no encuentra óptimo	Funciona pero con limitaciones	Implementación correcta que encuentra $\pi^*$
<b>Implementación NN</b>	Sin implementación	No sigue lógica NN correctamente	Implementa NN con errores menores	Implementación correcta del algoritmo NN
<b>Visualizaciones</b>	Sin visualizaciones	Figuras estáticas sin proceso	Visualizaciones básicas del proceso	Animaciones claras del proceso completo
<b>Análisis comparativo</b>	Sin análisis	Sin cálculo de gap o tiempos	Calcula gap y tiempos sin interpretación	Gap, tiempos y análisis fundamentado
<b>Código</b>	Sin código	Código no funcional o sin documentación	Código funcional pero poco documentado	Código estructurado, documentado, reproducible
<b>Redacción</b>	Ilegible o incomprendible	Múltiples errores graves de redacción	Redacción aceptable con errores menores	Redacción profesional y formato consistente

**Puntaje máximo documentación:** 24 puntos (8 criterios  $\times$  3 puntos)

### Presentación oral (50 % de la nota)

Criterio	No presentado (0)	Deficiente (1)	Aceptable (2)	Completo (3)
<b>Claridad expositiva</b>	No presenta o incomprensible	Exposición confusa y desorganizada	Exposición comprensible con saltos lógicos	Exposición clara, estructurada y fluida
<b>Dominio técnico</b>	Sin entendimiento	Poco dominio de conceptos	Entiende conceptos básicos	Dominio sólido, explica decisiones técnicas
<b>Visualizaciones</b>	No muestra visualizaciones	Muestra pero sin explicación	Muestra y explica parcialmente	Presenta y explica efectivamente
<b>Análisis crítico</b>	Sin análisis	Presenta resultados sin análisis	Análisis superficial	Análisis comparativo bien fundamentado
<b>Manejo del tiempo</b>	No respeta tiempo (<7 o >20 min)	Muy corta o larga (<8 o >18 min)	Duración aceptable con desbalances	Duración apropiada (10–15 min) equilibrada
<b>Respuestas a preguntas</b>	No puede responder	Respuestas incorrectas o evasivas	Responde con dudas o imprecisiones	Responde con seguridad y precisión
<b>Material de apoyo</b>	Sin apoyo visual	Diapositivas con errores o ilegibles	Material aceptable pero mejorable	Material profesional y efectivo
<b>Trabajo en equipo</b>	Desorganización evidente	Un solo integrante presenta	Participación desigual	Participación equilibrada del equipo

**Puntaje máximo presentación:** 24 puntos (8 criterios  $\times$  3 puntos)

**Nota final:** 
$$\text{Nota} = 1,0 + \left( \frac{\text{Puntaje documentación}}{24} \times 0,5 + \frac{\text{Puntaje presentación}}{24} \times 0,5 \right) \times 6,0$$

*Equivalente a:* 
$$\text{Nota} = 1,0 + \frac{\text{Puntaje total}}{48} \times 6,0$$