INSTITUTO TECNOLÓGICO NACIONAL DE MÉXICO CAMPUS CULIACÁN

TOPICOS DE IA

Alumno:

Portillo Zuñiga Steve Javier

Unidad 2 Tarea 1

Hora: 10:00 – 11:00

Maestro: Zuriel Dathan Mora Felix

Problema de Programación de Trabajos:

El Problema de Programación de Trabajos (JSSP - Job Shop Scheduling Problem) es un problema clásico de optimización en el área de la investigación de operaciones e inteligencia artificial. Se centra en programar un conjunto de trabajos en múltiples máquinas, respetando restricciones específicas.

Descripción del Problema

1. Definición del Problema:
    - Hay un conjunto de n trabajos J1,J2,...,JnJ_1, J_2, ..., J_nJ1,J2,...,Jn
    - Cada trabajo consta de una secuencia de m operaciones que deben procesarse en un conjunto de m máquinas M1,M2,...,MmM_1, M_2, ..., M_mM1,M2,...,Mm.
    - Cada operación debe ejecutarse en una máquina específica durante un tiempo determinado.
    - La secuencia de operaciones de cada trabajo es fija y debe respetarse.
2. Restricciones:
    - Secuencia fija: Dentro de un trabajo, las operaciones deben ejecutarse en el orden predefinido.
    - Máquina única: Una máquina solo puede procesar una operación a la vez.
    - No interrupciones: Una vez iniciada una operación, no puede ser interrumpida hasta que finalice.
    - Minimización del tiempo total: El objetivo suele ser minimizar el makespan, que es el tiempo total hasta que se finalizan todos los trabajos.

Representación del Problema

Supongamos que tenemos 3 máquinas (M1, M2, M3) y 3 trabajos (J1, J2, J3), y que cada trabajo tiene un conjunto de operaciones que deben realizarse en un orden específico.

Aquí hay una representación simple:

- J1: O1 -> M1 (3), O2 -> M2 (2), O3 -> M3 (2)
- J2: O1 -> M2 (2), O2 -> M1 (1), O3 -> M3 (4)

- J3: O1 -> M3 (4), O2 -> M2 (3), O3 -> M1 (3)

Notación

- O1, O2, O3: Operaciones del trabajo.
- M1, M2, M3: Máquinas.
- (n): Tiempo que tarda la operación en realizarse en la máquina correspondiente.

En esta representación, el objetivo es encontrar una secuencia en la que las operaciones se asignen a las máquinas, respetando el orden de las operaciones en cada trabajo y minimizando el makespan.

Solución

Una posible solución podría ser:

Tiempo	**M1**	**M2**	**M3**
0-3	J1-O1		
3-5		J1-O2	
5-7			J1-O3
7-8	J2-O2		
8-10			J2-O3
10-12		J2-O1	
12-15	J3-O3		
15-18		J3-O2	
18-22			J3-O1

Esta tabla muestra una posible secuencia de operaciones en las máquinas, donde hemos tratado de minimizar el makespan.

Problema de las N Reinas:

El Problema de las N Reinas es un problema clásico de inteligencia artificial y optimización combinatoria. Su objetivo es colocar N reinas en un tablero de ajedrez de N×N de manera que ninguna de ellas se ataque entre sí.

Descripción del Problema

1. Reglas del Problema:

    - Se debe colocar exactamente N reinas en el tablero.
    - Ninguna reina puede compartir la misma fila, columna o diagonal con otra.

2. Ejemplo para N = 8:
   Una posible solución para el tablero 8×8 es:

	A	B	C	D	E	F	G	H
1	Q	.	.	.	.	.	.	.
2	.	.	.	.	Q	.	.	.
3	.	.	.	.	.	.	Q	.
4	.	.	Q	.	.	.	.	.
5	.	.	.	.	.	.	.	Q
6	.	Q	.	.	.	.	.	.
7	.	.	.	Q	.	.	.	.
8	.	.	.	.	.	Q	.	.

Aquí, cada "Q" representa una reina, y cada "." representa un espacio vacío en el tablero.

Explicación del Algoritmo

Para encontrar esta solución, utilizamos el algoritmo de backtracking, que busca colocar las reinas una por una en cada fila del tablero y retrocede (backtrack) si encuentra una posición que no es segura.

1. Iniciamos con un tablero vacío de 8x8.
2. Intentamos colocar una reina en la primera fila (fila 0) en la primera columna (columna 0).
3. Verificamos si la posición es segura (es decir, si no hay otra reina en la misma columna, ni en la misma diagonal). Como el tablero está vacío, esta posición es segura.
4. Colocamos una reina en la posición (0, 0).
5. Avanzamos a la siguiente fila (fila 1) y repetimos el proceso.
6. Si encontramos una posición insegura, retrocedemos a la fila anterior y movemos la reina en la fila anterior a la siguiente columna posible.
7. Continuamos este proceso hasta que todas las filas tengan una reina colocada de manera segura.

Restricciones

- El valor de N debe ser un número entero positivo.
- El problema no tiene solución para N = 2 y N = 3.
- La solución debe garantizar que cada reina esté en una fila y columna únicas, sin compartir diagonales con otra reina.

Evaluación del Problema

Para evaluar la solución del problema de las N Reinas, se pueden utilizar los siguientes criterios:

1. Validez: Verificar que cada reina esté ubicada de manera que no se ataque con otra.
2. Eficiencia: Medir el tiempo de ejecución y la cantidad de retrocesos realizados en el algoritmo de backtracking.
3. Completitud: Confirmar que el algoritmo encuentra todas las soluciones posibles para un valor dado de N.
4. Escalabilidad: Analizar el rendimiento del algoritmo a medida que aumenta el valor de N.

Este problema es un buen ejemplo de búsqueda en espacios de estado y se puede resolver también con algoritmos de optimización como algoritmos genéticos o algoritmos basados en recocido simulado.

Árbol de Expansión Mínima:

El Árbol de Expansión Mínima (MST) es un concepto clave en teoría de grafos y algoritmos de optimización. Se aplica en problemas de conectividad óptima, como redes eléctricas, telecomunicaciones y planificación de rutas.

Descripción del Problema

Dado un grafo ponderado y conexo G=(V,E) donde:

- V es el conjunto de nodos (vértices).
- E es el conjunto de aristas (conexiones entre nodos), cada una con un peso asociado (coste).

El MST es un subconjunto de E que:

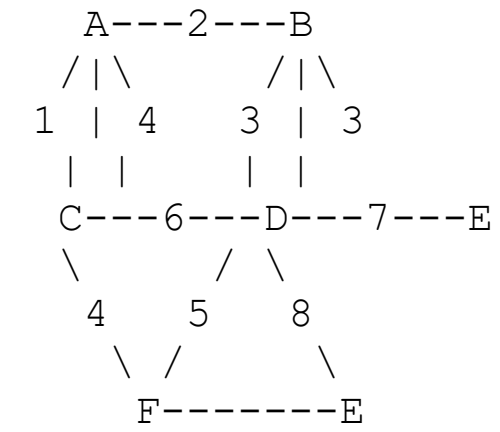
1. Conecta todos los nodos sin formar ciclos.
2. Minimiza la suma de los pesos de las aristas utilizadas.

Es decir, el árbol de expansión mínima conecta todos los nodos con el menor costo total posible.

Representación del Problema

Supongamos que tenemos un grafo con 6 nodos y los siguientes arcos con sus respectivos pesos:

Grafo:



Pasos para Encontrar el MST

Vamos a usar el algoritmo de Prim para encontrar el MST. Aquí están los pasos:

1. Seleccionar un nodo inicial: Comenzamos desde cualquier nodo, digamos el nodo A.
2. Elegir el arco con el menor peso: De A, el arco con el menor peso es A-C (peso 1).
3. Agregar el nodo conectado por el arco seleccionado al MST.
4. Elegir el arco con el menor peso desde cualquier nodo en el MST: El siguiente arco más pequeño es B-A (peso 2).
5. Repetir hasta que todos los nodos estén conectados.

Solucion

Algoritmo de Prim:

Paso	Nodo Actual	Arco Seleccionado	Peso	Nodos en MST	Arcos en MST	Peso Total
1	A	-	-	A	-	0
2	C	A-C	1	A, C	A-C	1
3	B	A-B	2	A, B, C	A-C, A-B	3

4	D	B-D	3	A, B, C, D	A-C, A-B, B-D	6
5	E	D-E	7	A, B, C, D, E	A-C, A-B, B-D, D-E	13
6	F	C-F	4	A, B, C, D, E, F	A-C, A-B, B-D, D-E, C-F	17

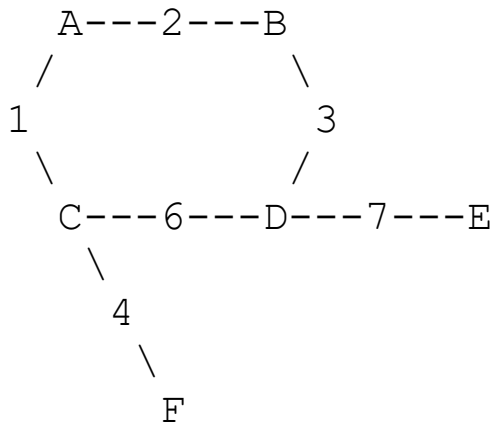
Resultado Final

El Árbol de Expansión Mínima (MST) resultante tiene los siguientes arcos:

- A-C (peso 1)
- A-B (peso 2)
- B-D (peso 3)
- D-E (peso 7)
- C-F (peso 4)

El peso total del MST es 17.

Y el grafo quedaría así



Restricciones

- El grafo debe ser conexo; de lo contrario, no es posible encontrar un MST que conecte todos los nodos.
- Los pesos de las aristas deben ser positivos.
- No se permiten ciclos en el MST.

Evaluación del Problema

Para evaluar la solución del MST, se pueden utilizar los siguientes criterios:

1. Correctitud: Verificar que el conjunto de aristas en el MST conecta todos los nodos sin formar ciclos.

2. Optimalidad: Asegurar que la suma de los pesos de las aristas seleccionadas es la menor posible.
3. Eficiencia: Medir el tiempo de ejecución del algoritmo, especialmente en grafos grandes.
4. Escalabilidad: Analizar el comportamiento del algoritmo a medida que aumenta el número de nodos y aristas en el grafo.

Problema del Agente Viajero

El Problema del Agente Viajero (Traveling Salesman Problem, TSP) es un problema clásico de optimización combinatoria y teoría de grafos. Se plantea de la siguiente manera:

Definición del Problema

El objetivo del TSP es encontrar la ruta más corta posible que permita a un agente (vendedor) visitar una serie de ciudades exactamente una vez y regresar a la ciudad de origen. La ruta debe minimizar la distancia total recorrida.

Representación del Problema

El problema se puede representar mediante un grafo completo, donde:

- Cada ciudad es un vértice del grafo.
- Cada par de ciudades está conectado por una arista (camino) con un peso asociado, que representa la distancia entre las dos ciudades.

Supongamos que tenemos las siguientes ciudades y distancias entre ellas:

	A	B	C	D	E
A	0	10	15	20	25
B	10	0	35	25	30
C	15	35	0	30	20
D	20	25	30	0	15
E	25	30	20	15	0

En este grafo, las ciudades A, B, C, D y E están conectadas por caminos con las distancias correspondientes.

Soluciones al Problema

El TSP es un problema NP-completo, lo que significa que no existe un algoritmo eficiente conocido que pueda encontrar la solución óptima en tiempo polinómico para todos los casos posibles. Sin embargo, existen varios enfoques para resolverlo:

1. Fuerza Bruta: Consiste en evaluar todas las permutaciones posibles de rutas y seleccionar la de menor costo. Es exacto pero computacionalmente costoso.
2. Heurísticas y Aproximaciones:

    - Algoritmo del Vecino Más Cercano (Nearest Neighbor): Comienza en una ciudad y elige iterativamente la ciudad más cercana no visitada.
    - Algoritmo de Inserción: Inserta las ciudades en la ruta de manera iterativa, optimizando la distancia total en cada paso.
    - Algoritmo de Enfriamiento Simulado (Simulated Annealing): Utiliza un enfoque probabilístico para encontrar soluciones aproximadas.
    - Algoritmo Genético: Basado en principios de evolución y selección natural, encuentra soluciones aproximadas mediante la generación de poblaciones de rutas.

Ejemplo con el Algoritmo del Vecino Más Cercano

Para ilustrar una solución, usaremos el algoritmo del vecino más cercano, comenzando desde la ciudad A:

1. Desde A, la ciudad más cercana es B (10 unidades).
2. Desde B, la ciudad más cercana no visitada es D (25 unidades).
3. Desde D, la ciudad más cercana no visitada es E (15 unidades).
4. Desde E, la ciudad más cercana no visitada es C (20 unidades).
5. Finalmente, regresamos a A desde C (15 unidades).

Ruta Resultante

La ruta resultante es: A -> B -> D -> E -> C -> A

Cálculo de la Distancia Total

Camino	Distancia
A -> B	10
B -> D	25
D -> E	15
E -> C	20
C -> A	15
Total	85

La distancia total recorrida es 85 unidades.

Aplicaciones del TSP

El problema del agente viajero tiene numerosas aplicaciones prácticas, incluyendo:

- Logística y Distribución: Optimización de rutas de entrega para minimizar costos y tiempo.
- Planificación de Rutas de Vehículos: Diseño de rutas eficientes para servicios de transporte.
- Fabricación: Minimización del tiempo de movimiento de herramientas y materiales en plantas de producción.
- Turismo: Creación de itinerarios óptimos para turistas que desean visitar múltiples destinos.