Informe: Solución al Problema de Seguridad en el Museo

Franco Hernández Piloto, Hivan Cañizares Diaz

December 2024

1 Introducción y Definición del Problema

En este informe, abordamos el problema de optimización para la seguridad de un museo. El objetivo principal es **ubicar estratégicamente dispositivos de seguridad** en una habitación del museo para **vigilar todos los objetos de valor** dentro de ella, **minimizando el costo total** de los dispositivos utilizados debido a un presupuesto limitado.

Se consideran los siguientes tipos de dispositivos de seguridad:

1. Sensor de Proximidad: Detecta la presencia en áreas específicas.
2. Cámara de Vigilancia de 360 Grados: Vista panorámica completa sin puntos ciegos.
3. Cámara de Vigilancia de 120 Grados: Amplio campo de visión para áreas grandes.
4. Cámara de Vigilancia de 60 Grados: Vigilancia detallada para objetos específicos.

Restricción de Ubicación de Dispositivos en Vértices: Una simplificación crucial que hemos adoptado en nuestra formulación del problema es la restricción de ubicar los dispositivos de seguridad exclusivamente en los **vértices** del polígono que representa la habitación del museo. Esta decisión se fundamenta en varias consideraciones prácticas y de complejidad algorítmica:

- **Reducción de la Complejidad del Problema:** Permitir la colocación de dispositivos en cualquier punto dentro del polígono o incluso en los bordes incrementaría significativamente la complejidad del problema de optimización. La búsqueda de ubicaciones óptimas en un espacio continuo o incluso a lo largo de las aristas del polígono resultaría en un problema mucho más difícil de modelar y resolver computacionalmente. Al restringir las ubicaciones a un conjunto discreto de puntos (los vértices), transformamos el problema en uno de naturaleza combinatoria, más manejable y

susceptible a ser abordado con técnicas como la reducción a Weighted Set Cover.

- **Justificación Práctica:** En muchos escenarios reales, los vértices de una habitación (esquinas, columnas, etc.) suelen ser ubicaciones lógicas y convenientes para la instalación de dispositivos de seguridad. Colocar dispositivos en estos puntos estratégicos permite una cobertura efectiva del espacio interior. Además, desde un punto de vista de infraestructura, suele ser más sencillo y estético instalar dispositivos en esquinas o puntos de unión de paredes.

- **Cobertura Adecuada:** Aunque restringir las ubicaciones a los vértices podría no siempre conducir a la solución absolutamente óptima en un sentido continuo, en la práctica, una selección adecuada de dispositivos en los vértices puede proporcionar una cobertura de vigilancia suficientemente buena para la mayoría de las habitaciones de museo. La clave está en elegir los tipos de dispositivos apropiados y, en el caso de cámaras con ángulo de visión limitado, considerar múltiples orientaciones desde cada vértice, como hemos propuesto con la discretización de direcciones.

- **Transformación a un Problema Discreto:** La restricción a los vértices nos permite discretizar el espacio de posibles soluciones. En lugar de buscar en un espacio continuo de ubicaciones, ahora solo debemos decidir qué tipo de dispositivo (o ninguno) colocar en cada vértice. Esta discretización es fundamental para poder formular el problema como un Weighted Set Cover y aplicar algoritmos de optimización combinatoria.

Si bien somos conscientes de que esta restricción puede limitar el espacio de soluciones y potencialmente excluir la solución óptima global en un sentido irrestricto, consideramos que es un **compromiso razonable y necesario** para hacer el problema abordable y obtener soluciones prácticas y efectivas para la seguridad del museo. Sin esta simplificación, el problema se volvería significativamente más complejo y difícil de resolver con las herramientas de optimización disponibles.

Por tanto el problema, en su forma inicial, se puede resumir como:

- **Entrada:**
  - Plano de la habitación del museo representado como un polígono simple.
  - Coordenadas 2D de los vértices del polígono.
  - Coordenadas 2D de cada objeto de valor dentro del polígono.
  - Tipos de dispositivos de seguridad disponibles donde se especifica el rango de visibilidad de cada uno y sus costos.

- **Salida:**

- Una configuración de dispositivos de seguridad (tipo y ubicación en los vértices del polígono) que garantice la vigilancia de todos los objetos de valor.
- Esta configuración debe tener el costo total mínimo posible.

2 Enfoque Propuesto: Reducción al Problema de Weighted Set Cover

Nuestro enfoque para resolver este problema consiste en **reducirlo al problema clásico de Weighted Set Cover (Cobertura de Conjuntos Ponderada)**. El problema de Weighted Set Cover es un problema de optimización combinatoria bien conocido y estudiado, para el cual existen algoritmos de aproximación y técnicas de solución.

La idea central es modelar el problema de seguridad en el museo en términos de conjuntos y costos, de manera que encontrar una solución óptima para el Weighted Set Cover resultante nos proporcione una solución óptima para el problema de seguridad.

3 Reducción a Weighted Set Cover

Definimos los siguientes elementos:

3.1 Universo (U)

Sea U el **conjunto de todos los objetos de valor** en el museo. Cada objeto de valor es un elemento único en este universo.

- **Definición:** $U = \{o_1, o_2, ..., o_n\}$, donde o_i representa el i-ésimo objeto de valor en el museo, y n es el número total de objetos de valor. Aquí, o_1, o_2, hasta o_n representan los objetos individuales en el conjunto U.

3.2 Colección de Conjuntos (S) y sus Pesos

Para cada vértice v del polígono que representa la habitación del museo, y para cada tipo de dispositivo de seguridad d disponible, definimos un conjunto S_{vd}. Este conjunto contiene todos los objetos de valor que son **vigilados**(más abajo se define que es ser vigilado,por ahora es mejor abstraerse de eso) por el dispositivo d si este se coloca en el vértice v.

- **Definición:** Para cada vértice v del polígono y cada tipo de dispositivo d en {Sensor de Proximidad, Cámara 360°, Cámara 120°, Cámara 60°}, definimos el conjunto S_{vd} como: $S_{vd} = \{o \in U \mid$ el objeto o es vigilado por el dispositivo d colocado en el Esto significa que S_{vd} es el conjunto de todos los objetos o que pertenecen al universo U (objetos de valor) y que son vigilados por el dispositivo d cuando se coloca en el vértice v.

Cada conjunto S_{vd} tiene un peso asociado, que corresponde al costo de instalar el dispositivo d.

- **Definición:** Sea costo(d) el costo de un dispositivo de tipo d. El peso del conjunto S_{vd} se define como: peso(S_{vd}) = costo(d). En otras palabras, el peso del conjunto S_{vd} es simplemente el costo del dispositivo d.

3.3 Instancia del Problema Weighted Set Cover

Con las definiciones anteriores, podemos construir una instancia del problema Weighted Set Cover de la siguiente manera:

- **Universo:** $U = \{o_1, o_2, ..., o_n\}$ (Conjunto de objetos de valor).
- **Colección de Conjuntos:** $S = \{S_{vd} \mid$ para cada vértice v del polígono y cada tipo de dispositivo $d\}$. Esto significa que S es una colección de todos los conjuntos S_{vd} que podemos formar considerando cada vértice v y cada tipo de dispositivo d.
- **Pesos de los Conjuntos:** Para cada S_{vd} en S, el peso es peso(S_{vd}) = costo(d).

3.4 Reducción

El problema de seguridad en el museo se reduce al problema Weighted Set Cover de la siguiente manera:

- **Instancia del problema de Seguridad:** Polígono, vértices, objetos de valor, tipos de dispositivos y sus costos.
- **Instancia del problema Weighted Set Cover (construida a partir de la instancia de Seguridad):**
  - Universo U = Conjunto de objetos de valor.
  - Colección de conjuntos $S = \{S_{vd}\}$ (donde S_{vd} son los objetos vigilados por el dispositivo d en el vértice v).
  - Pesos de los conjuntos: peso(S_{vd}) = costo(d).

Resolver el problema Weighted Set Cover para esta instancia nos dará una solución que puede traducirse directamente a una solución para el problema de seguridad en el museo:

- **Solución de Weighted Set Cover:** Un subconjunto de conjuntos C contenido en S tal que la unión de todos los conjuntos en C es igual a U (es decir, la unión de todos los S_{vd} que están en C es igual a U) y la suma de los pesos de los conjuntos en C es mínima.

- **Solución del problema de Seguridad:** Para cada conjunto S_{vd} en C (la solución de Weighted Set Cover), colocamos el dispositivo d en el vértice v,sin importar si se ubican más de un dispositivo en un mismo vértice,esto no afecta ya que esta configuración de dispositivos garantizará que todos los objetos de valor estén vigilados (porque la unión de los conjuntos cubre U) y tendrá el costo mínimo (porque la solución de Weighted Set Cover minimiza la suma de los pesos).

3.5 Correctitud de la Reducción

Demostración:

- **Si encontramos una solución óptima para Weighted Set Cover:** Esta solución nos indica qué conjuntos S_{vd} seleccionar. Cada selección de S_{vd} corresponde a colocar un dispositivo d en el vértice v. Dado que la solución de Weighted Set Cover cubre todo el universo U (todos los objetos), sabemos que todos los objetos estarán vigilados. Además, como la solución de Weighted Set Cover es de costo mínimo, la configuración de dispositivos resultante también tendrá el costo mínimo.
- **Si existe una solución óptima para el problema de Seguridad:** Esta solución consiste en una configuración de dispositivos en los vértices que vigila todos los objetos con el menor costo posible. Podemos mapear esta solución a una solución de Weighted Set Cover seleccionando los conjuntos S_{vd} correspondientes a los dispositivos colocados en cada vértice. Dado que la solución de seguridad cubre todos los objetos, la unión de estos conjuntos cubrirá el universo U. Y como la solución de seguridad es de costo mínimo, la solución de Weighted Set Cover resultante también tendrá un peso mínimo.

4 Pseudocódigo de Algoritmo de Construcción del Weighted Set Cover

Definición de Objeto Vigilado: un objeto es vigilado por un dispositivo si y solo si sus coordenadas se encuentran dentro del rango de detección y en el caso de las camaras, si se encuentra dentro del ángulo de dirección a la que apunta y no hay ninguna pared obstruyendo la visión de dicho objeto.

```
   Función ConstruirWeightedSetCover(poligono, vertices, objetos, tipos_de_dispositivos,
costos_de_dispositivos):

   // -- Inicialización --
   Universo U = conjunto de todos los objetos // U = {objeto1, objeto2,
..., objetoN}
   ColeccionDeConjuntos S = conjunto vacío
```

```
   PesosDeConjuntos = diccionario vacío // Para almacenar pesos asociados
a cada conjunto

   // Para cada vértice posible para colocar dispositivos
   Para cada vertice v en vertices:
      // -- Sensores de Proximidad --
      conjunto_proximidad = ObtenerObjetosEnRangoProximidad(vertice
v, objetos, radio_proximidad)
      nombre_conjunto_proximidad = "SensorProximidad_Vertice" + indice(v)
// Nombre único para el conjunto
      AgregarConjunto(S, conjunto_proximidad)
      PesosDeConjuntos[nombre_conjunto_proximidad] = costo_sensor_proximidad

      // -- Cámaras de 360 Grados --
      conjunto_360 = ObtenerObjetosVistosPorCamara360(vertice v,objetos)
//Por cada objeto se verifica si la recta que hay entre la camara y
el objeto no intercepta con ningun lado del poligono,lo que significa
que no hay ninguna pared obstruyendo la vision desde la camara al objeto.
      nombre_conjunto_360 = "Camara360_Vertice" + indice(v)
      AgregarConjunto(S, conjunto_360)
      PesosDeConjuntos[nombre_conjunto_360] = costo_camara_360

      // -- Cámaras de 120 y 60 Grados --
      direcciones_discretizadas = {0, 5, 10, ..., 355} // Direcciones
cada 5 grados

      Para cada direccion_camara en direcciones_discretizadas:
         // -- Cámara de 120 Grados --
         conjunto_120 = ObtenerObjetosVistos(vertice v, direccion_camara,
objetos, tipo_camara = "120 grados")
         nombre_conjunto_120 = "Camara120_Vertice" + indice(v) + "_Dir"
+ direccion_camara
         Si conjunto_120 no está vacío:  // Opcional:  Solo agregar
conjuntos no vacíos para eficiencia
            AgregarConjunto(S, conjunto_120)
            PesosDeConjuntos[nombre_conjunto_120] = costo_camara_120

         // -- Cámara de 60 Grados --
         conjunto_60 = ObtenerObjetosVistos(vertice v, direccion_camara,
objetos, tipo_camara = "60 grados")
         nombre_conjunto_60 = "Camara60_Vertice" + indice(v) + "_Dir"
+ direccion_camara
         Si conjunto_60 no está vacío:  // Opcional:  Solo agregar conjuntos
no vacíos para eficiencia
            AgregarConjunto(S, conjunto_60)
            PesosDeConjuntos[nombre_conjunto_60] = costo_camara_60
```

```
   // Retornar la instancia del Weighted Set Cover
   Retornar Universo U, ColeccionDeConjuntos S, PesosDeConjuntos

   // Funciónes auxiliares

   Función ObtenerObjetosEnRangoProximidad(vertice, objetos, radio):
      objetos_en_rango = conjunto vacío
      Para cada objeto o en objetos:
         Si distancia(vertice, objeto o) <= radio:
            Agregar objeto o a objetos_en_rango
      Retornar objetos_en_rango

   Función ObtenerObjetosVistos(vertice_camara, direccion_camara, objetos,
tipo_camara):
      objetos_visibles = conjunto vacío
      Para cada objeto o en objetos:
         Si ObjetoEsVisibleDesdeCamara(vertice_camara, direccion_camara,
objeto o, tipo_camara):
            Agregar objeto o a objetos_visibles
      Retornar objetos_visibles

   Función ObjetoEsVisibleDesdeCamara(vertice_camara, direccion_camara,
objeto, tipo_camara):
```

- Sea $c = (x_c, y_c)$ las coordenadas del `vertice_camara`.
- Sea $o = (x_o, y_o)$ las coordenadas del `objeto` o.
- Sea α_{camara} la `direccion_camara` en grados.
- Determinar el ángulo de visión θ_{vision} basado en `tipo_camara`:
  - Si `tipo_camara` es "120 grados", $\theta_{vision} = 120°$.
  - Si `tipo_camara` es "60 grados", $\theta_{vision} = 60°$.
- Calcular el vector desde la cámara al objeto: $\vec{co} = (x_o - x_c, y_o - y_c)$.
- Si $\vec{co} = (0, 0)$, retornar `Verdadero` (objeto en la misma posición que la cámara).
- Convertir la `direccion_camara` α_{camara} de grados a radianes: $\alpha_{camara\_rad} =$ `radians`(α_{camara}).
- Calcular el vector de dirección de la cámara: $\vec{d} = (\cos(\alpha_{camara\_rad}), \sin(\alpha_{camara\_rad}))$.
- Calcular el producto punto entre $\vec{co}$ y $\vec{d}$: $dot\_product = \vec{co} \cdot \vec{d} = (x_o - x_c)\cos(\alpha_{camara\_rad}) + (y_o - y_c)\sin(\alpha_{camara\_rad})$.
- Calcular la magnitud del vector $\vec{co}$: $magnitude\_co = ||\vec{co}|| = \sqrt{(x_o - x_c)^2 + (y_o - y_c)^2}$.

- Calcular el coseno del ángulo entre $\vec{co}$ y $\vec{d}$:

$$\cos(\theta) = \frac{dot\_product}{magnitude\_co}$$

- Calcular el coseno del ángulo mitad del campo de visión: $cos\_half\_fov = \cos(\texttt{radians}(\theta_{vision}/2))$.
- Si $\cos(\theta) \geq cos\_half\_fov$, entonces el objeto está dentro del campo de visión angular de la cámara.

Puntos Clave del Pseudocódigo:

  - **Claridad de la Abstracción:** El pseudocódigo se enfoca en la lógica de construcción del Weighted Set Cover, utilizando las funciones abstractas `ObtenerObjetosVistos()` y `ObtenerObjetosEnRangoProximidad()`. No se detalla *cómo* se implementan estas funciones internamente.
  - **Generación de Conjuntos y Pesos:** Muestra claramente cómo se crean los conjuntos $S_{v,d}$ (o $S_{v,d,\alpha}$) para cada tipo de dispositivo y cada vértice (y orientación, para las cámaras de ángulo fijo), y cómo se asignan los pesos correspondientes (costos de los dispositivos).
  - **Instancia de Weighted Set Cover:** Al final del proceso, se obtiene la instancia del Weighted Set Cover: el universo U, la colección de conjuntos S, y los pesos. Esta instancia puede ser alimentada a un algoritmo de Weighted Set Cover.

5 Análisis de Complejidad y Correctitud del Algoritmo de Construcción del Weighted Set Cover

5.1 Complejidad del Algoritmo `ConstruirWeightedSetCover`

El algoritmo propuesto para construir la instancia del problema Weighted Set Cover, `ConstruirWeightedSetCover`, tiene una complejidad que puede ser analizada de la siguiente manera:

  - **Número de Vértices (V):** Sea V el número de vértices del polígono del museo.
  - **Número de Objetos de Valor (N):** Sea N el número de objetos de valor dentro del museo.
  - **Número de Direcciones Discretizadas (D):** Sea D el número de direcciones discretizadas consideradas para las cámaras de 120° y 60°. En nuestro caso, con incrementos de 5 grados, $D = 72$.

- **Inicialización:** La creación del conjunto `U` y la inicialización de `S` y `PesosDeConjuntos` toma tiempo constante o lineal en el número de objetos, que podemos denotar como $O(N)$, donde N es el número de objetos de valor.
- **Bucle Externo (vértices):** El bucle principal itera sobre cada vértice del polígono. Sea V el número de vértices del polígono. Este bucle se ejecuta V veces.
- **Dentro del Bucle de Vértices:**
  - **Sensor de Proximidad:** `ObtenerObjetosEnRangoProximidad` itera sobre todos los objetos y calcula la distancia para cada uno. Si el cálculo de distancia es $O(1)$, entonces esta función toma $O(N)$ tiempo.
  - **Cámara de 360 Grados:** `ObtenerObjetosVistosPorCamara360` también iteraría sobre todos los objetos y realizaría una comprobación de visibilidad.La comprobación de visibilidad para un objeto toma en el peor caso $O(E)$ donde E es el número de aristas del polígono (para verificar intersecciones con cada arista), `ObtenerObjetosVistosPorCamara360` tomaría $O(N \cdot E)$ tiempo. En un polígono simple, $E \approx V$, por lo que podría ser $O(N \cdot V)$.
  - **Cámaras de 120° y 60°:**
    - **Bucle de Direcciones Discretizadas:** Tenemos un conjunto de direcciones discretizadas. Si usamos incrementos de 5 grados desde 0° a 355°, hay $360/5 = 72$ direcciones. Denotemos el número de direcciones discretizadas como D. En este caso, $D = 72$.
    - `ObtenerObjetosVistos`: Para cada dirección discretizada y cada tipo de cámara (120° y 60°), llamamos a `ObtenerObjetosVistos`. Dentro de `ObtenerObjetosVistos`, iteramos sobre todos los objetos y realizamos cálculos geométricos para determinar la visibilidad. Aunque los cálculos geométricos para saber si el objeto se encuentra dentro del ángulo de la dirección actual se hacen en O(1), el cálculo de visibilidad en base a si hay alguna pared obstruyendo la vista es similar al caso de la Cámara de 360 Grados,entonces `ObtenerObjetosVistos` toma $O(N \cdot V)$ tiempo.
- **Complejidad Total Aproximada:**
  - Para cada vértice, hacemos:
    - $O(N)$ para Sensor de Proximidad.
    - $O(N \cdot V)$ para Cámara de 360°.
    - Para cada dirección discretizada (D direcciones) y para dos tipos de cámaras (120° y 60°), hacemos $O(N \cdot V)$ por cada `ObtenerObjetosVistos`. Esto es aproximadamente Ø(2 $\cdot D \cdot$ $(N \cdot V)$ = O(D $\cdot N \cdot V$ pero D en realidad es considerada

una constante ya que es un valor que se escoje al analizar el tamaño de los objetods del museo,luego quedaría Ø(N $\cdot V$.

* Sumando todo para un vértice: $O(N) + 2 \cdot O(N \cdot V) = O(N + 2 \cdot N \cdot V) = O(N \cdot V)$.
* Como esto se repite para cada vértice (V vértices), la complejidad total aproximada para construir la instancia de Weighted Set Cover sería: $O(N \cdot V \cdot V)$

La construcción de la instancia de Weighted Set Cover tiene una complejidad polinomial en el número de vértices, el número de objetos y el número de direcciones discretizadas.

5.2 Correctitud y Discretización de Direcciones

- **Margen de Error por Discretización:** Es cierto que al discretizar las direcciones cada 5 grados, introducimos un margen de error. Existe la posibilidad teórica de que un objeto esté justo en el borde del campo de visión de una cámara en una dirección que *no* coincide exactamente con una de nuestras direcciones discretizadas. En este caso, si las direcciones discretizadas más cercanas no "ven" al objeto, podríamos perder la oportunidad de cubrir ese objeto con esa cámara en esa posición.
- **Asunción sobre el Tamaño de los Objetos:** Se asume que el tamaño de los objetos es lo suficientemente grande. Si los objetos no son puntuales, sino que tienen un área significativa, y si el incremento angular entre direcciones discretizadas (5 grados) es pequeño en comparación con el tamaño angular típico de los objetos vistos desde un vértice, entonces es **muy probable** que al menos una de las direcciones discretizadas capture una porción significativa del objeto, y por lo tanto, consideremos que el objeto está "vigilado".
- **Densidad de Direcciones:** Discretizar cada 5 grados es una elección de compromiso. Si quisiéramos reducir aún más el margen de error, podríamos usar incrementos más pequeños (e.g., 1 grado o incluso menos). Sin embargo, esto aumentaría el número de direcciones discretizadas D y el tamaño de la instancia de Weighted Set Cover.
- **Correctitud *dentro del marco discretizado*:** Dentro de las limitaciones impuestas por la discretización, podemos argumentar que **la estrategia es correcta en el sentido de que explora sistemáticamente un conjunto representativo de posibles configuraciones de vigilancia.** Para cada vértice y cada tipo de dispositivo, consideramos:
  * Sensor de Proximidad: Cubre objetos cercanos.
  * Cámara 360°: Cubre todos los objetos visibles dependiendo de las obstrucciones(paredes).

* Cámaras de 120° y 60°: Exploramos una serie de direcciones discretizadas que, en conjunto, proporcionan una cobertura angular razonablemente densa alrededor de cada vértice.

Al construir el Weighted Set Cover a partir de estos conjuntos, estamos efectivamente creando un conjunto de "opciones de cobertura" que representan las formas más relevantes de vigilar los objetos desde los vértices, dadas las limitaciones de los dispositivos y la discretización.

5.3 Demostración de "Correctitud" (Relativa a la Discretización)

Dado el uso de direcciones discretizadas, no podemos garantizar una solución *absolutamente* óptima en el espacio continuo de todas las posibles orientaciones de cámara. Sin embargo, podemos decir que:

- **Exhaustividad Discretizada:** Para cada vértice y tipo de cámara de ángulo fijo, el algoritmo considera un conjunto de orientaciones discretizadas que cubren razonablemente el espacio angular alrededor del vértice.
- **Cobertura Completa (dentro de las opciones discretizadas):** Si existe una solución que utiliza cámaras de ángulo fijo en los vértices para vigilar todos los objetos, y si las orientaciones requeridas para estas cámaras están "suficientemente cerca" de alguna de nuestras direcciones discretizadas (dentro del margen de error aceptable debido a la discretización y el tamaño de los objetos), entonces nuestro algoritmo considerará opciones de cobertura que son *representativas* de esa solución.

6 Demostración de la NP-Completitud del Problema de Weighted Set Cover (Cobertura de Conjuntos Ponderada)

La demostración se divide en dos partes: primero, se muestra que el problema está en NP, y segundo, se demuestra que es NP-duro mediante una reducción polinomial desde el problema de Set Cover.

7 Introducción

El problema de Weighted Set Cover es una generalización del problema de Set Cover clásico. En el problema de Set Cover, el objetivo es minimizar

el número de conjuntos seleccionados para cubrir un universo dado. En Weighted Set Cover, cada conjunto tiene un costo asociado, y el objetivo es minimizar el costo total de los conjuntos seleccionados que cubren el universo.

Formalmente, el problema de Weighted Set Cover se define como sigue:

[Problema de Weighted Set Cover] Dado un universo $U = \{u_1, u_2, \ldots, u_n\}$, una colección de subconjuntos de U, $S = \{S_1, S_2, \ldots, S_m\}$, donde cada $S_i \subseteq U$, y una función de costo $c : S \to \mathbb{Q}^+$ que asigna un costo positivo racional a cada conjunto en S. El problema de Weighted Set Cover consiste en encontrar una subcolección $S' \subseteq S$ tal que:

1. $\bigcup_{S_i \in S'} S_i = U$ (Cobertura: S' cubre el universo U).
2. $\sum_{S_i \in S'} c(S_i)$ es minimizado (Minimización del costo total).

En este documento, demostraremos que el problema de decisión asociado al problema de optimización de Weighted Set Cover es NP-completo. El problema de decisión se formula de la siguiente manera:

[Problema de Decisión de Weighted Set Cover] Dado un universo U, una colección de subconjuntos S de U, una función de costo $c : S \to \mathbb{Q}^+$, y un valor $K \in \mathbb{Q}^+$. ¿Existe una subcolección $S' \subseteq S$ tal que $\bigcup_{S_i \in S'} S_i = U$ y $\sum_{S_i \in S'} c(S_i) \leq K$?

8 NP-Completitud de Weighted Set Cover

Para demostrar que el problema de decisión de Weighted Set Cover es NP-completo, debemos mostrar dos cosas:

1. Weighted Set Cover está en NP.
2. Weighted Set Cover es NP-duro.

8.1 Weighted Set Cover está en NP

Para demostrar que Weighted Set Cover está en NP, debemos mostrar que si se nos da una solución candidata, podemos verificar en tiempo polinomial si es una solución válida y si cumple la condición de costo.

Supongamos que se nos da una instancia del problema de decisión de Weighted Set Cover: (U, S, c, K), y una solución candidata $S' \subseteq S$. Necesitamos verificar en tiempo polinomial si S' es una cobertura válida y si el costo total no excede K.

1. **Verificación de cobertura:** Para verificar si S' es una cobertura de U, debemos comprobar si $\bigcup_{S_i \in S'} S_i = U$. Podemos hacer esto iterando sobre cada elemento $u \in U$ y verificar si existe al menos un

conjunto $S_i \in S'$ tal que $u \in S_i$. Si para cada $u \in U$ encontramos tal S_i, entonces S' es una cobertura. Esta verificación puede realizarse en tiempo polinomial en el tamaño de la entrada (tamaño de U y S).

2. **Verificación del costo total:** Para verificar si el costo total de S' no excede K, debemos calcular $\sum_{S_i \in S'} c(S_i)$ y compararlo con K. Podemos sumar los costos de los conjuntos en S' y comparar la suma con K. Esta operación también puede realizarse en tiempo polinomial en el tamaño de la entrada.

Dado que ambas verificaciones pueden realizarse en tiempo polinomial, el problema de decisión de Weighted Set Cover está en NP.

8.2 Weighted Set Cover es NP-duro

Para demostrar que Weighted Set Cover es NP-duro, reduciremos el problema de Set Cover (que se sabe que es NP-completo) al problema de Weighted Set Cover en tiempo polinomial.

Realizaremos una reducción polinomial desde el problema de Set Cover al problema de Weighted Set Cover.

Consideremos una instancia del problema de Set Cover: (U, S), donde $U = \{u_1, u_2, \ldots, u_n\}$ es el universo y $S = \{S_1, S_2, \ldots, S_m\}$ es una colección de subconjuntos de U. El problema de Set Cover consiste en determinar si existe una subcolección $S' \subseteq S$ de tamaño a lo sumo k tal que $\bigcup_{S_i \in S'} S_i = U$. El problema de decisión de Set Cover es: ¿Existe una cobertura de conjuntos de tamaño a lo sumo k?

Ahora, construimos una instancia del problema de Weighted Set Cover a partir de la instancia de Set Cover. Definimos la instancia de Weighted Set Cover como (U, S, c, K), donde:

- El universo U y la colección de subconjuntos S son los mismos que en la instancia de Set Cover.
- Definimos la función de costo $c : S \to \mathbb{Q}^+$ tal que para cada $S_i \in S$, $c(S_i) = 1$. Es decir, asignamos un costo de 1 a cada conjunto en S.
- Establecemos el valor límite de costo $K = k$.

Ahora, debemos demostrar que la instancia de Set Cover (U, S) tiene una solución de tamaño a lo sumo k si y solo si la instancia de Weighted Set Cover $(U, S, c, K = k)$ tiene una solución con costo total a lo sumo $K = k$.

1. **Si la instancia de Set Cover tiene una solución de tamaño a lo sumo k, entonces la instancia de Weighted Set Cover tiene una solución con costo total a lo sumo k.** Supongamos que existe una subcolección $S'_{SC} \subseteq S$ de tamaño $|S'_{SC}| \leq k$ que es una cobertura para la instancia de Set Cover (U, S). Consideremos

la misma subcolección $S'_{WSC} = S'_{SC}$ para la instancia de Weighted Set Cover $(U, S, c, K = k)$.

- S'_{WSC} es una cobertura de U porque S'_{SC} es una cobertura de U.
- El costo total de S'_{WSC} es $\sum_{S_i \in S'_{WSC}} c(S_i) = \sum_{S_i \in S'_{SC}} 1 = |S'_{SC}| \leq k = K$.

Por lo tanto, S'_{WSC} es una solución válida para la instancia de Weighted Set Cover con costo total a lo sumo $K = k$.

2. **Si la instancia de Weighted Set Cover tiene una solución con costo total a lo sumo k, entonces la instancia de Set Cover tiene una solución de tamaño a lo sumo k.** Supongamos que existe una subcolección $S'_{WSC} \subseteq S$ que es una cobertura para la instancia de Weighted Set Cover $(U, S, c, K = k)$ y que el costo total es $\sum_{S_i \in S'_{WSC}} c(S_i) \leq k$. Consideremos la misma subcolección $S'_{SC} = S'_{WSC}$ para la instancia de Set Cover (U, S).
   - S'_{SC} es una cobertura de U porque S'_{WSC} es una cobertura de U.
   - El tamaño de S'_{SC} es $|S'_{SC}| = |S'_{WSC}|$. El costo total de S'_{WSC} es $\sum_{S_i \in S'_{WSC}} c(S_i) = \sum_{S_i \in S'_{WSC}} 1 = |S'_{WSC}|$. Por lo tanto, $|S'_{SC}| = |S'_{WSC}| = \sum_{S_i \in S'_{WSC}} c(S_i) \leq k$.

   Por lo tanto, S'_{SC} es una solución válida para la instancia de Set Cover de tamaño a lo sumo k.

Hemos demostrado que existe una solución para el problema de Set Cover de tamaño a lo sumo k si y solo si existe una solución para el problema de Weighted Set Cover con costo total a lo sumo k, utilizando la reducción descrita. La reducción es polinomial porque la construcción de la instancia de Weighted Set Cover a partir de la instancia de Set Cover implica simplemente asignar un costo de 1 a cada conjunto y establecer $K = k$, lo cual se puede hacer en tiempo polinomial.

Dado que Set Cover es NP-completo y hemos encontrado una reducción polinomial desde Set Cover a Weighted Set Cover, concluimos que Weighted Set Cover es NP-duro.

9 NP-Completitud de Set Cover

Para demostrar que el problema de decisión de Set Cover es NP-completo, debemos mostrar dos cosas:

1. Set Cover está en NP.
2. Set Cover es NP-duro.

9.1 Set Cover está en NP

[Demostración de que Set Cover está en NP] Supongamos que se nos da una instancia del problema de decisión de Set Cover: (U, S, k), y una solución candidata $S' \subseteq S$. Necesitamos verificar en tiempo polinomial si S' es una cobertura válida y si su tamaño no excede k.

1. **Verificación de cobertura:** Para verificar si S' es una cobertura de U, debemos comprobar si $\bigcup_{S_i \in S'} S_i = U$. Podemos hacer esto iterando sobre cada elemento $u \in U$ y verificar si existe al menos un conjunto $S_i \in S'$ tal que $u \in S_i$. Si para cada $u \in U$ encontramos tal S_i, entonces S' es una cobertura. Esta verificación puede realizarse en tiempo polinomial en el tamaño de la entrada.
2. **Verificación del tamaño:** Para verificar si el tamaño de S' no excede k, simplemente contamos el número de conjuntos en S', $|S'|$, y comparamos con k. Esta operación también puede realizarse en tiempo polinomial.

Dado que ambas verificaciones pueden realizarse en tiempo polinomial, el problema de decisión de Set Cover está en NP.

9.2 Set Cover es NP-duro

Para demostrar que Set Cover es NP-duro, reduciremos el problema de Vertex Cover (que se asume que es NP-completo) al problema de Set Cover en tiempo polinomial.

[Demostración de que Set Cover es NP-duro] Realizaremos una reducción polinomial desde el problema de Vertex Cover al problema de Set Cover.

Consideremos una instancia del problema de Vertex Cover: $G = (V, E)$ y un entero k. El problema de Vertex Cover consiste en determinar si existe un conjunto de vértices $V' \subseteq V$ de tamaño a lo sumo k tal que para cada arista $(u, v) \in E$, al menos uno de los vértices u o v está en V'.

Ahora, construimos una instancia del problema de Set Cover a partir de la instancia de Vertex Cover. Definimos la instancia de Set Cover como (U, S, k), donde:

- El universo U es el conjunto de aristas E del grafo G. Es decir, $U = E$.
- Para cada vértice $v \in V$, creamos un conjunto $S_v \subseteq U$ que contiene todas las aristas incidentes al vértice v en G. Es decir, $S_v = \{e \in E \mid v \text{ es un extremo de } e\}$. Definimos la colección de conjuntos $S = \{S_v \mid v \in V\}$.
- El valor límite k es el mismo que en la instancia de Vertex Cover.

Ahora, debemos demostrar que la instancia de Vertex Cover (G, k) tiene una solución de tamaño a lo sumo k si y solo si la instancia de Set Cover $(U = E, S = \{S_v\}_{v \in V}, k)$ tiene una solución de tamaño a lo sumo k.

1. **Si la instancia de Vertex Cover tiene una solución de tamaño a lo sumo k, entonces la instancia de Set Cover tiene una solución de tamaño a lo sumo k.** Supongamos que existe un vertex cover $V' \subseteq V$ de tamaño $|V'| \leq k$ para el grafo G. Consideremos la subcolección de conjuntos $S' = \{S_v \mid v \in V'\} \subseteq S$. El tamaño de S' es $|S'| = |V'| \leq k$. Debemos demostrar que S' es una cobertura de $U = E$. Para cualquier arista $e = (u, w) \in E$, como V' es un vertex cover, al menos uno de los extremos de e, digamos u (o w), debe estar en V'. Si $u \in V'$, entonces e es una arista incidente a u, por lo que $e \in S_u$. Como $S_u \in S'$, tenemos que $e \in \bigcup_{S_v \in S'} S_v$. Esto se cumple para cualquier arista $e \in E$, por lo tanto, $\bigcup_{S_v \in S'} S_v = E = U$. Así, S' es una cobertura de tamaño a lo sumo k.
2. **Si la instancia de Set Cover tiene una solución de tamaño a lo sumo k, entonces la instancia de Vertex Cover tiene una solución de tamaño a lo sumo k.** Supongamos que existe una subcolección $S' \subseteq S$ de tamaño $|S'| \leq k$ que es una cobertura de $U = E$. Como cada conjunto en S es de la forma S_v para algún $v \in V$, podemos escribir $S' = \{S_{v_1}, S_{v_2}, \ldots, S_{v_l}\}$ donde $l = |S'| \leq k$ y $v_1, v_2, \ldots, v_l \in V$. Consideremos el conjunto de vértices $V'' = \{v_1, v_2, \ldots, v_l\}$. El tamaño de V'' es $|V''| = l \leq k$. Debemos demostrar que V'' es un vertex cover para G. Para cualquier arista $e = (u, w) \in E = U$, como S' es una cobertura de U, debe existir al menos un conjunto en S' que contenga e. Supongamos que $e \in S_{v_i}$ para algún $S_{v_i} \in S'$. Por definición de S_{v_i}, esto significa que v_i es un extremo de $e = (u, w)$. Por lo tanto, ya sea $v_i = u$ o $v_i = w$ (o ambos). En cualquier caso, al menos uno de los extremos de e (a saber, v_i) está en $V'' = \{v_1, v_2, \ldots, v_l\}$. Esto se cumple para cualquier arista $e \in E$, por lo tanto, V'' es un vertex cover de tamaño a lo sumo k.

Hemos demostrado que la instancia de Vertex Cover (G, k) tiene una solución de tamaño a lo sumo k si y solo si la instancia de Set Cover $(U = E, S = \{S_v\}_{v \in V}, k)$ tiene una solución de tamaño a lo sumo k. La reducción es polinomial porque la construcción del universo $U = E$ y la colección de conjuntos $S = \{S_v\}_{v \in V}$ a partir del grafo $G = (V, E)$ se puede realizar en tiempo polinomial en el tamaño de G. Para cada vértice, iteramos sobre sus aristas incidentes para construir el conjunto correspondiente.

Dado que Vertex Cover es NP-completo y hemos encontrado una reducción polinomial desde Vertex Cover a Set Cover, concluimos que Set Cover es NP-duro.

10 Solución

10.1 Algoritmo Greedy para la Cobertura de Conjuntos Ponderada

El algoritmo greedy para la cobertura de conjuntos ponderada construye una cobertura eligiendo repetidamente un conjunto S que minimice el peso w_s dividido por el número de elementos en S aún no cubiertos por los conjuntos elegidos. Se detiene y devuelve los conjuntos elegidos cuando forman una cobertura.

10.2 Pseudocódigo

```
GreedySetCover(S, U, w):
    Entrada:
        S: Una colección de conjuntos
        U: Los elementos a ser cubiertos
        w: Una función de pesos de los conjuntos de S
    Output:
        C: Un subconjunto de S que cubre U

C ← {}
covered ← {}

    while covered  U do
        best_set ← null
        min_cost_per_element ←

        for each set s in S do
            uncovered_elements ← s \ covered  // Elements in s that are not yet covered

            if uncovered_elements is not empty then
                cost_per_element ← w(s) / |uncovered_elements|

                if cost_per_element < min_cost_per_element then
                    min_cost_per_element ← cost_per_element
                    best_set ← s

        C ← C  best_set
        covered ← covered  best_set

    return C
```

10.3 Teorema de Aproximación

Sea H_k denotado como $\sum_{i=1}^{k}(1/i) \approx \ln(k)$, donde k es el tamaño del conjunto más grande.

El algoritmo greedy devuelve una cobertura de conjuntos de peso a lo sumo H_k veces el peso mínimo de cualquier cobertura.

10.4 Prueba del Teorema

Cuando el algoritmo greedy elige un conjunto s, imagine que carga el precio por elemento para esa iteración a cada elemento recién cubierto por s. Entonces, el peso total de los conjuntos elegidos por el algoritmo es igual a la cantidad total cargada, y cada elemento se carga una vez.

Considere cualquier conjunto $S = \{x_k, x_{k-1}, \ldots, x_1\}$ en la cobertura de conjuntos óptima C^*. Sin pérdida de generalidad, suponga que el algoritmo greedy cubre los elementos de s en el orden dado: $x_k, x_{k-1}, \ldots, x_1$. Al inicio de la iteración en la que el algoritmo cubre el elemento x_i de s, al menos i elementos de s permanecen sin cubrir. Por lo tanto, si el algoritmo greedy eligiera s en esa iteración, pagaría un costo por elemento de a lo sumo w_s/i. Por lo tanto, en esta iteración, el algoritmo greedy paga a lo sumo w_s/i por elemento cubierto. Por lo tanto, se le cobra al elemento x_i a lo sumo w_s/i por ser cubierto. Sumando sobre i, la cantidad total cargada a los elementos en S es a lo sumo $w_s H_k$. Sumando sobre $s \in C^*$ y notando que cada elemento está en algún conjunto en C^*, la cantidad total cargada a los elementos en general es a lo sumo $\sum_{s \in C^*} w_s H_k = H_k OPT$.

10.5 Complejidad Temporal

La complejidad temporal del algoritmo greedy de cobertura de conjuntos depende del número de conjuntos, el tamaño del universo y el esfuerzo requerido para calcular la relación costo-beneficio en cada paso. Los pasos de inicialización (crear la cobertura vacía y el conjunto de elementos cubiertos) toman un tiempo de $O(1)$. El bucle "while" itera hasta que todos los elementos estén cubiertos. En el peor de los casos, esto podría requerir considerar cada conjunto en S múltiples veces. Dentro del bucle, el algoritmo itera a través de cada conjunto s en S para encontrar aquel con el mínimo costo por elemento no cubierto. Esto toma un tiempo de $O(|S|)$. Para cada conjunto, determinar los elementos no cubiertos implica comparar los elementos en el conjunto con los elementos ya cubiertos, lo que toma un tiempo de $O(|U|)$ en el peor de los casos. Añadir el mejor conjunto a la cobertura y actualizar el conjunto de elementos cubiertos toma un tiempo de $O(|U|)$. Combinando estos factores, la complejidad temporal general puede estimarse de la siguiente manera:

- El bucle exterior se ejecuta como máximo $|U|$ veces.

- El bucle interior se ejecuta $|S|$ veces.
- La computación dentro del bucle interior toma un tiempo de $O(|U|)$.

Por lo tanto, la complejidad temporal total es $O(|U||S||U|) = O(|U|^2|S|)$. Esto puede ser una consideración significativa para universos grandes y colecciones de conjuntos grandes.

10.6 Desventajas

El algoritmo puede que no siempre encuentre la solución óptima, pero por el teorema planteado se puede garantizar que la solución es a lo máximo $\ln(n)$ de la solución óptima, lo cuál también puede ser un factor muy grande, especialmente para valores pequeños de n. El algoritmo también depende mucho del orden inicial de los conjuntos, lo cual puede afectar la calidad de la solución.