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.

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 de la Correctitud de la Reducción:

- **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
```

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.