Segundo Problema de DAA

Carla Sunami Pérez Valera

Septiembre 2024

1 Enunciado del Problema

Problema de Monitoreo de Velocidad

No es broma cuando decimos que los accidentes automovilísticos se han vuelto cada vez más frecuentes en nuestro día a día. La continua violación de las leyes de tránsito o las carreras ilegales son solo algunas de las causas de estos incidentes.

Ahora bien, se conoce que en una calle de longitud n metros, los conductores tienen la manía de sobrepasar el límite de velocidad señalado. Puesto que esta calle es una zona delicada, ya que es muy usada por niños, ancianos y el público en general, se quiere tener total control de que esto no ocurra.

Para velar por el cumplimiento estricto del límite de velocidad, la alcaldía se ha dado a la tarea de instalar a lo largo de la calle cámaras de vigilancia con sensores de velocidad integrados, de modo tal que ni un metro de la calle quede sin ser monitoreado, para atrapar a cada conductor que infrinja la regla antes mencionada. Pero, obviamente, la alcaldía desea usar la mínima cantidad de recursos para llevar a cabo esta tarea y, quién sabe, tal vez ampliar este proyecto urbano a otra calle con el mismo problema.

Las cámaras que se instalarán tienen un rango de visibilidad (el cual es el mismo en ambos sentidos de la calle). Es decir, una cámara, una vez puesta en su poste, es capaz de supervisar la calle m metros hacia ambos lados. Por ejemplo, si ponemos una cámara a 7 metros del inicio de la calle y el rango de monitoreo es de 3 metros, entonces esa cámara monitoreará desde el 4º metro hasta el 10º metro de la calle (contando desde el inicio). **Nota:** Las cámaras cerrán ubicadas en postes ya existentes para minimizar así el gasto de recursos.

2 Modelo del Problema

Consideremos una calle de longitud n metros, donde los conductores suelen sobrepasar el límite de velocidad. La alcaldía ha decidido instalar cámaras de vigilancia con un rango de visibilidad m metros en ambos sentidos. Esto significa que una cámara instalada en un poste p_i puede monitorear desde $p_i - m$ hasta $p_i + m$.

Para ilustrar el problema, supongamos que tenemos k postes distribuidos a lo largo de la calle. La distribución de los postes se puede representar como:

$$1 \dots p_1 \dots (2m{+}1) \quad (2m{+}1){+}1 \dots p_2 \dots 2(2m{+}1) \quad \cdots \quad (k{-}1)(2m{+}1){+}1 \dots p_k \dots k(2m{+}1)$$

La distribución de los postes se organiza de tal manera que sus rangos no se solapen y que no quede ningún espacio libre. En este caso, la cantidad mínima de cámaras necesarias para cubrir la calle es k, uno por cada poste.

2.1 Argumento del Adversario

Sea ADV un adversario que afirma tener una solución al problema sin revisar al menos un bombillo. Si se le proporciona a ADV la distribución de los postes, existen dos casos posibles:

- Si ADV dice que no es un alumbramiento correcto, se equivoca, ya que la distribución de los postes alumbra la calle completamente.
- Si ADV dice que sí es un alumbramiento correcto, supongamos que b_i es la cámara que no revisó. Si se le da la distribución D' (que es D sin b_i), su respuesta no cambiaría y sería incorrecta, ya que el intervalo $[p_i - m, p_i + m]$ solo era alumbrado por la cámara b_i.

2.1.1 Conclusión

La técnica del adversario demuestra que es imposible determinar la viabilidad de una distribución de cámaras sin revisar al menos una cámara, ya que cualquier respuesta dada por el adversario puede ser incorrecta si no se considera la cantidad mínima de cámaras necesarias. Por lo tanto, la cota mínima de cámaras necesarias para cubrir completamente la calle es igual al número de postes, es decir, k.

La complejidad del algoritmo sería O($|postes|$**) .**

3 Distribucion Valida

Interpretaremos una calle de n metros de longitud como un arreglo de n elementos. De esta forma consideramos que colocar una cámara en la posición i es alumbrar los 2m metros adyacentes a i, es decir, marcar con 1 el intervalo [i m, i + m]. Entonces una combinación satisface encender la calle completamente si cada posición del arreglo (cada metro de la misma) contiene un 1 luego de colocar todas las cámaras de la combinación. En un primer paso descartamos el hecho de que aún colocando cámaras en todos los postes, no sea posible monitorear la calle completa. Esto se logra comprobando el siguiente lema: Lema 1 Una calle de longitud n, con postes ubicados en las posiciones p1, p2, . . . , pk y cumpliendo que 1 p1 ¡ p2 ¡ . . . ¡ pk n, es monitoreable con caámaras de alcance m si y solo si:

• El primer poste aparece a una distancia menor o igual que m metros del inicio de la calle: p1 m 1
• El último poste aparece a una distancia menor o igual que m metros del final de la calle: pk + m n
• Entre cada par de postes hay a lo sumo 2m metros de distancia, es decir i, j tal que i ¡ j: pj pi 2m + 1
Cada uno de los subconjuntos de la distribución, si cumple el Lema 1, constituye una Distribución Válida. **La solución será la menor cantidad de cámaras que se requieran para generar una Distribución Válida.**

4 Solución

4.1 Fruerza Bruta

La idea intuitiva q se nos ocurre es generar todas las posibles combinaciones de seleccionar los postes que serán encendidos y quedarnos con aquella que monitoree toda la calle, utilizando la menor cantidad de cámaras. Para ello utilizamos una función generadora de cadenas binarias de tamaño cantidad de postes, donde cada 1 de la cadena representa colocar una camara en dicha posición.

4.1.1 Análisis de la Complejidad Temporal

Consideremos una instancia del problema donde se tiene como entrada una calle de longitud n, los postes $p_1, p_2, \ldots, p_k$ tales que $1 \leq p_1 < p_2 < \ldots < p_k \leq n$ y bombillos con rango m. El algoritmo sigue los siguientes pasos:

1. Comprobar que la calle sea alumbrable con la distribución inicial de los postes, utilizando el método `check(n, postes, m)` de complejidad temporal $O(|postes|)$.

2. Si se cumple el paso 1, asignar $best = |postes|$, que es una operación de tiempo constante $O(1)$.

3. Sea A el conjunto potencia de postes. Comprobar para cada conjunto $c \in A$ el cumplimiento del Lema 1 con el método `check(n, c, m)`. La complejidad temporal de este paso es $O(|postes| \cdot 2^{|postes|})$.

4. Devolver *best* como respuesta, que es otra operación de tiempo constante $O(1)$.

La complejidad total del algoritmo es:

$$O(|postes|) + O(1) + O(|postes| \cdot 2^{|postes|}) + O(1) = O(|postes| \cdot 2^{|postes|})$$

4.2 Programación Dinámica

tengase en cuenta q iré perfeccionando la solucion a medida q va avanzando este documento

4.2.1 Programacion Dinámica sin optimizar

Toda instancia del problema, donde hay una calle de n metros, un conjunto de postes $p_1, p_2, \ldots, p_k$ tales que $1 \leq p_1 < p_2 < \ldots < p_k \leq n$, con cámaras de alcance m, puede ser interpretado como el siguiente problema T:

Dado un arreglo de tamaño n y un conjunto de intervalos:

$$[p_1 - m, p_1 + m], [p_2 - m, p_2 + m], \ldots, [p_k - m, p_k + m]$$

Determinar cuál es la menor cantidad de intervalos necesaria para cubrir el arreglo de tamaño n.

Lema 2

Existe una biyección entre T y el problema inicial, es decir, una combinación de elementos de $p_1, p_2, \ldots, p_k$ que solucione el problema Monitoreo de Velocidad (a partir de ahora MV) es una solución de T, y viceversa.

Demostración Lema 2

Notar que cada intervalo de T coincide con el rango de iluminación de un poste y hay la misma cantidad de postes que intervalos. Por lo que a cada intervalo $[p_i - m, p_i + m]$ de T se le puede asociar el poste p_i en MV, es decir, el rango de iluminación posible de p_i coincide con el intervalo $[p_i - m, p_i + m]$.

Sea S una solución de T, sea P_0 el conjunto de intervalos escogidos para lograr la solución, se cumple que $|P_0| = S$. Escoger todo $p_i \in P_0$ es un alumbramiento correcto de una calle de n metros. Si no es solución de AC es porque existe un conjunto de postes P_0' que es un monitoreo correcto y $|P_0'| < S$, siendo el conjunto de intervalos asociados a P_0' un cubrimiento completo del arreglo con una cantidad menor que S, lo que no es posible ya que S es solución de T. Entonces se cumple que si S es solución de T implica que también lo es de MV.

Sea S' una solución de MV, sea P_1 el conjunto de postes escogidos para lograr la solución, se cumple que $|P_1| = S'$. Escoger todo $p_i \in P_1$ es un cubrimiento correcto del arreglo con los intervalos $[p_i - m, p_i + m]$. Si no es solución de T es porque existe un conjunto de intervalos P_1' que es un cubrimiento correcto y $|P_1'| < S'$, siendo el conjunto de postes asociados a P_1' un monitoreo correcto con una cantidad menor que S', lo que no es posible ya que S' es solución de MV. Entonces se cumple que si S' es solución de MV también lo es de T.

La siguiente solución presenta una estrategia dinámica. Sea dp un arreglo de tamaño n que cumple que:

$$dp[i] contienelamenorcantidaddeintervalosnecesariosparacubrirelarreglodesdelaposición 1 hastalai.$$

En términos del problemaMonitoreo de Velocidad equivale a decir que:

$$dp[i] contienelamenorcantidaddebombillosnecesariosparaalumbrarlacalledesdelaposición 1 hastalai.$$

Entonces la solución del problema está en $dp[n]$, ya que sería la cantidad mínima de cámaras necesarias para iluminar el camino desde el inicio hasta n contando con todos los postes disponibles.

Correctitud

Lema 3 En el problema T para todo par de intervalos consecutivos $[p_k - m, p_k + m], [p_{k+1} - m, p_{k+1} + m]$ se cumple que en cada uno hay al menos una posición que no está en el otro.

Demostración Lema 3 Como $p_k < p_{k+1}$ se cumple que $p_k - m < p_{k+1} - m$, por tanto $p_k - m \notin [p_{k+1} - m, p_{k+1} + m]$. Igual se cumple que $p_k + m < p_{k+1} + m$, por lo que $p_{k+1} + m \notin [p_k - m, p_k + m]$.

Lema 4 Sea el conjunto de intervalos $[p_1 - m, p_1 + m], [p_2 - m, p_2 + m], \dots, [p_k - m, p_k + m]$. Si una posición x no es alcanzable por la izquierda (derecha) del intervalo I_i, es decir, $x < p_i - m$ $(p_i + m < x)$, entonces tampoco puede ser alcanzada por ningún intervalo I_j donde $i < j (j < i)$.

Demostración Lema 4 Sea x una posición cualquiera, sea $I_i = [p_i - m, p_i + m]$ el primer intervalo tal que x no es alcanzable por su izquierda (derecha), entonces $x < p_i - m$ $(p_i + m < x)$. Aplicando inducción en j para los intervalos $I_{i+j}(I_{i-j})$:

Si $j = 1$: Entonces $I_{i+j} = I_{i+1}(I_{i-j} = I_{i-1})$, es el intervalo que está inmediatamente después (antes) de I_i. Por el Lema 3 en el intervalo I_i va a existir una posición t que no está en $I_{i+1}(I_{i-1})$, por tanto $t < I_{i+1}(I_{i-1} < t)$ y como I_i no alcanza a x por la izquierda (derecha) se cumple que $x < t (t < x)$, entonces:

$$x < t < I_{i+1}(I_{i-1} < t < x)$$

Entonces x tampoco es alcanzable por la izquierda (derecha) de $I_{i+1}(I_{i-1})$.

Supongamos que se cumple para $j = k$: Si $I_{i+k}(I_{i-k})$ lo cumple entonces $x < I_{i+k}(I_{i-k} < x)$, aplicando el Lema 3, en I_i existe un t que no está en $I_{i+k}(I_{i-k})$ tal que $t < I_{i+k+1}(I_{i-(k+1)} < t)$ y como $x < t(t < x)$ entonces:

$$x < t < I_{i+k+1}(I_{i-(k+1)} < t < x)$$

Entonces x tampoco es alcanzable por la izquierda (derecha) de $I_{i+k+1}(I_{i-(k+1)})$.

Queda demostrado por inducción que se cumple $\forall j$ con $i < j (j < i)$.

Lema 5 En el problema T un intervalo $[p_i - m, p_i + m]$, donde i indica que es el i-ésimo intervalo de los k existentes, puede tener la opción de no utilizarse en el cubrimiento si y solo si se cumple alguno de los siguientes puntos:

1. Si es el primer intervalo se tiene que cumplir que $p_2 - m \leq 1$, es decir que el segundo cubra hasta el inicio del arreglo.
2. Si es el último intervalo se tiene que cumplir que $p_{k-1} + m \geq n$, es decir que el penúltimo cubra hasta el final del arreglo.
3. Si es un intervalo intermedio entonces se tiene que cumplir que $(p_{i+1} - m) - (p_{i-1} + m) \leq 1$, es decir que los intervalos adyacentes a él se solapen o sean continuos.

Demostración Lema 5 Si se cumple alguno de los tres puntos es fácil ver que el intervalo puede no utilizarse, ya que el área que él cubriría en el arreglo puede ser cubierta por otro intervalo.

Demostrando que en caso contrario no puede dejar de utilizarse: sea $[p_i - m, p_i + m] un intervalo de los k$ existentes:

1. Si es el primer intervalo y se cumple que $p_2 - m > 1$, entonces el intervalo $R = [1, p_2 - m - 1]$ no puede ser cubierto por I_2, y por el Lema 4 tampoco podrá serlo por I_j con $2 < j$. Entonces el único intervalo capaz de cubrir a R es I_1, teniendo que estar presente obligatoriamente en el cubrimiento ya que tiene que cumplirse el Lema 1.

2. Si es el último intervalo (I_k) y se cumple que $p_{k-1} + m < n$, entonces el intervalo $R = [p_{k-1} + m + 1, n]$ no puede ser cubierto por I_{k-1}, y por el Lema 4 tampoco podrá serlo por I_j con $j < k - 1$. Entonces el único intervalo capaz de cubrir a R es I_k, teniendo que estar presente obligatoriamente en el cubrimiento ya que tiene que cumplirse el Lema 1.

3. Si es un intervalo intermedio y se cumple que $(p_{i+1} - m) - (p_{i-1} + m) > 1$, hay al menos una posición x tal que $I_{i-1} < x < I_{i+1}$. Entonces x no es alcanzable por la derecha ni la izquierda de I_{i-1} e I_{i+1} respectivamente, por el Lema 4 se puede afirmar entonces que $\forall j$ donde $j \leq i - 1$ o $i + 1 \leq j$ el intervalo I_j no puede contener a x. Como se cumple Lema 1, y todo intervalo distinto de I_i no contiene a x, necesariamente I_i pertenece al cubrimiento.

Queda demostrado por el contrarrecíproco que si el intervalo I_i puede dejar de usarse entonces cumple uno de los tres puntos.

Lema 6 El arreglo dp es no decreciente.

Demostración Lema 6 La definición del arreglo plantea que $dp[i]$ contiene la menor cantidad de cámaras necesarios para alumbrar la calle desde la posición 1 hasta la i. Suponiendo que $\exists i, j$ con $i < j$ tal que $dp[i] > dp[j]$, sea c la combinación de postes que resuelve $dp[j]$. Por la propia definición de dp, dicha combinación monitorea desde el inicio hasta j y por tanto también monitorea desde el inicio hasta i.

Entonces $dp[i]$ no contiene la menor cantidad de cámaras necesarias para monitorear la calle desde el inicio hasta i, llegando a una contradicción con la propia definición de dp. Por reducción al absurdo se cumple que $\forall i, j$ con $1 \leq i < j \leq n$ se cumple que $dp[i] \leq dp[j]$.

Código de la Función

A continuación se presenta la implementación de la función ‘solve‘:

```
def solve(n, postes, m):
    if not check(n, postes, m):
        return -1
    dp = [0] * (n + 1)
    for i in range(0, len(postes)):
        temp = 1
        if postes[i] - m - 1 >= 0:
            temp = dp[postes[i] - m - 1] + 1
```

```
9           for j in range(max(1, postes[i] - m), min(
                postes[i] + m, n) + 1):
10              if dp[j] == 0:
11                  dp[j] = temp
12      return dp[n]
```

Descripción del Algoritmo

En un primer momento, en las líneas 2 y 3 se comprueba que la distribución de postes dada permita alumbrar la calle completamente, a través del método `check(n, postes, m)` verificando el cumplimiento del Lema 1.

De la línea 5 a la 11 se realiza un recorrido por todos los postes de izquierda a derecha modificando el arreglo dp como se explica a continuación:

Para cada poste p_k se recorre el intervalo $I_k = [p_k - m, p_k + m]$. Por cada posición $i \in I_k$, si $dp[i]$ no está definido (es igual a 0) entonces $dp[i] = dp[p_k - m - 1] + 1$, es decir, incrementar en 1 el valor que hay en la primera posición no alcanzable desde p_k hacia la izquierda, donde $dp[p_k - m - 1] = 0$ si $p_k - m - 1 \leq 1$.

Suponiendo que la entrada dada cumple con el Lema 1, demostraremos por inducción que en cada iteración k del algoritmo, el arreglo dp se encuentra bien definido desde el inicio hasta la posición $p_k + m$.

Caso Base: $k = 1$ Como se cumple el Lema 1, el intervalo $[1, p_1 + m]$ puede ser monitoreado por una sola cámara si se pone en el primer poste. Como 1 es la mejor solución posible para poder monitorear cualquier intervalo se cumple que dp está bien definido desde el inicio hasta $p_1 + m$ luego de la primera iteración.

Suposición Inductiva Suponiendo que se cumple $\forall m \leq k$, sea la iteración $k + 1$ del algoritmo, se cumple que dp está bien definido desde el inicio hasta $p_k + m$.

Sea $R = [p_k + m + 1, p_{k+1} + m]$ el intervalo de dp que debe ser definido en la presente iteración. Se cumple que $\forall i \in R, dp[i] \geq dp[p_{k+1} - m - 1]$ ya que dp es no decreciente por el Lema 6.

¿Podría $dp[i]$ ser igual a $dp[p_{k+1} - m - 1]$ para alguna $i \in R$? Ninguna de las combinaciones de postes que son solución del problema hasta la posición $p_{k+1} - m - 1$ puede contener a p_{k+1}, ya que este no alcanza a ninguna posición del intervalo $[1, p_{k+1} - m - 1]$ y no tiene sentido contarlo. Entonces el poste más cercano a R y que pudiera participar en dichas soluciones es p_k, pero p_k solo alcanza hasta $p_k + m$ y R comienza en $p_k + m + 1$, por lo que R no es alcanzable por la derecha de p_k y por el Lema 4 tampoco podrá hacerlo ningún poste antes de p_k. Entonces no hay forma de que una combinación que resuelva hasta $p_{k+1} - m - 1$ sea una solución para algún $i \in R$.

¿Puede ser $dp[p_{k+1} - m - 1] + 1$ una solución para todo $dp[i]$ con $i \in R$? Sea c una combinación cualquiera que resuelve $dp[p_{k+1} - m - 1]$, como $p_{k+1} \notin c$ y $R \subseteq [p_{k+1} - m, p_{k+1} + m]$, se puede afirmar que $c \cup \{p_{k+1}\}$ resuelve $dp[i]$ $\forall i \in R$, siendo entonces $dp[i] = dp[p_{k+1} - m - 1] + 1$.

Entonces $dp[p_{k+1} - m - 1] + 1$ es la cantidad mínima de postes necesaria para resolver $dp[i]$ $\forall i \in R$ y es resuelta en la iteración $k + 1$.

4.2.2 Complejidad Temporal

El algoritmo sigue los siguientes pasos:

1. Comprobar que la distribución de postes dada pueda monitorear toda la calle, igual que en la solución anterior, haciendo uso de `check(n, postes, m)`, con complejidad temporal $O(|postes|)$.
2. Se realizan $|postes|$ iteraciones tal que en la k-ésima iteración $dp[i]$ está correctamente definido $\forall i$ tal que $1 \leq i \leq p_k + m$, basándose en la demostración anterior. Complejidad temporal $O(2m \cdot |postes|)$.
3. Devolver $dp[n]$ como resultado del problema.

Entonces su complejidad temporal es:

$$O(|postes|) + O(2m \cdot |postes|) = O(2m \cdot |postes|).$$

4.3 Solución 3: Dinámica $O(n)$

La siguiente solución está basada en la anterior pero con una pequeña modificación en el código. Como se muestra a continuación:

```
def dinamica_2(n, postes, m):
    if not check(n, postes, m):
        return -1
    last = 0
    dp = [0] * (n + 1)
    for i in range(0, len(postes)):
        temp = 1
        if postes[i] - m - 1 >= 0:
            temp = dp[postes[i] - m - 1] + 1
        for j in range(max(last + 1, postes[i] - m), min(postes[i] + m, n) + 1):
            dp[j] = temp
        last = min(postes[i] + m, n)
    return dp[n]
```

Correctitud

La diferencia con la solución anterior es que, en este caso, en cada iteración k, se puede saber dónde comienza el intervalo R que tiene que ser definido en dp, que es $last + 1$. Esto se cumple debido a que en el final de cada iteración i se guarda en $last$ el valor $p_i + m$ o n en caso de que llegue al final (demostrado en la solución anterior).

Complejidad Temporal

El algoritmo sigue los siguientes pasos:

1. Comprobar que la distribución de postes dada pueda monitorear toda la calle, igual que en la solución anterior, haciendo uso de `check(n, postes, m)`, con complejidad temporal $O(|postes|)$.

2. Se realizan $|postes|$ iteraciones, en cada una de ellas se define en dp el intervalo R asociado que no ha sido visitado aún en el algoritmo y una vez que es definido no se vuelve a visitar, por lo que cada posición en dp es visitada una única vez. Como $|dp| = n$, la complejidad temporal sería $O(n)$.

3. Devolver $dp[n]$ como resultado del problema.

Entonces su complejidad temporal es:

$$O(|postes|) + O(n)$$

Como $|postes|$ es a lo sumo n, por el principio de la suma es:

$$O(n)$$

4.4 Solución 4: Dinámica $O(|postes|)$

Tomando como experiencia las soluciones 2 y 3, en la iteración $k + 1$ del algoritmo, la solución para el intervalo $R = [p_k + m + 1, p_{k+1} + m]$ se basa en la solución para la posición $p_{k+1} - m - 1$. Basado en esto, la presente solución resultó de proponer los siguientes objetivos:

1. En la iteración $k + 1$ poder saber el valor de $p_{k+1} - m - 1$.

2. En la iteración k no tener que definir en dp las posiciones del intervalo R asociado, sino actualizar la información almacenada de forma tal que la iteración $k + 1$ cumpla con el objetivo 1.

3. Si cada una de las iteraciones cumple con el objetivo 2, al final de las $|postes|$ iteraciones se puede saber cuál sería la solución para la posición n.

Lema 7

Para toda posición i de dp, se cumple que $dp[i - 1] = dp[i]$ o $dp[i] = dp[i + 1]$ y en caso de que no suceda es porque en ninguna otra posición j con $j \neq i$ se cumple que $dp[j] = dp[i]$. Es decir, todas las posiciones con igual valor en dp están en un único intervalo donde todos los valores son iguales.

Demostración Lema 7

Suponiendo que existen i, j, k tal que $dp[i] = dp[j] \neq dp[k]$, con $i < k < j$, pero esta distribución provocaría el incumplimiento del Lema 6.

Lema 8

Sea p_i la posición del i-ésimo poste, entonces en el intervalo $[p_i - m, p_i + m]$ solo puede existir uno o dos valores distintos en dp, donde m es la intensidad de los bombillos. En caso de que existan dos valores distintos a, b, con $a < b$, se tiene que cumplir que $a + 1 = b$.

Demostración Lema 8

Suponiendo que existe un poste p_i tal que en el intervalo $I_i = [p_i - m, p_i + m]$ existen más de dos valores distintos. Existen entonces dos posiciones $j, k \in I_i$ tales que $dp[j] < dp[k] - 1$, donde, por el Lema 6, $j < k$.

Sea S el conjunto de todas las combinaciones de postes que resuelven $dp[j]$, se cumple por la definición de dp que $\forall c \in S, |c| = dp[j]$, contemplamos los siguientes casos:

- Si existe un $c \in S$ que use el poste p_i entonces c alumbrará a los intervalos $[1, j]$ e I_i, como $j \in I_i$ ambos intervalos se interceptan, por lo que la combinación c alumbrará el intervalo $[1, j] \cup I_i = [1, p_i + m]$, por tanto a $[1, k]$ también, existe entonces una combinación de cardinalidad $dp[j]$ que resuelve $[1, k]$ con menos de $dp[k]$ bombillos, entrando en contradicción con la definición de dp ya que $dp[k]$ no sería la mejor solución.

- Si no existe un $c \in S$ que use el poste p_i entonces toda combinación en S más el poste p_i alumbrarí a el intervalo $[1, p_i + m]$ y por tanto a $[1, k]$. Entonces $[1, k]$ puede ser resuelto con $dp[j] + 1$ bombillos, llegando a una contradicción con la definición de dp ya que $dp[j] + 1 < dp[k]$ y $dp[k]$ no sería la mejor solución.

Queda demostrado entonces por reducción a lo absurdo que se cumple el Lema 8.

```
def dinamica_3(n, postes, m):
    if not check(n, postes, m):
        return -1
    A = [0] * (len(postes) + 1)
    A[1] = min(n, postes[0] + m)
    current = 1
    if A[1] == n:
        return 1
    for i in range(1, len(postes)):
        if A[max(0, current - 1)] < postes[i] - m - 1:
            current += 1
        A[current] = min(n, postes[i] + m)
        if A[current] == n:
            return current
    return -1
```

Correctitud

La idea del algoritmo es la siguiente:

1. Comprobar el cumplimiento del Lema 1 en las líneas 2-3.

2. En caso de que se cumpla 1 en las líneas 4-8 se crea el arreglo A que en la posición i contiene cuál es la mayor distancia desde el inicio de la calle que puede ser iluminada con i bombillos como mínimo. Se inicializa con el primer bombillo y se comprueba que sea solución. Además se declara la variable *current* que indica la cantidad de bombillos utilizados hasta el momento.

3. En las líneas 9-14 hay un ciclo en el cual, en cada iteración $k+1$, con $k = 0 \ldots |postes| - 1$, *current* va a ser igual a la cantidad mínima de postes para resolver el intervalo $R = [p_k + m + 1, p_{k+1} + m]$ asociado a la iteración y actualizando A como $A[current] = postes[k+1] + m$, además la primera vez que se cumpla que $A[current] = n$ se devuelve *current*.

En esta solución, se puede saber a través del arreglo A, cuáles serían los valores de dp en las soluciones 1 y 2 como se muestra a continuación:

$$dp[i] = t \iff A[t-1] < i \leq A[t]$$

Por el Lema 6 se cumple que dp es no decreciente, y en Lema 8 se afirma que cada intervalo asociado a un poste puede aumentar en 1 como máximo la cantidad de postes de la solución, por lo que se puede afirmar que toda posición i del arreglo dp cumple que $dp[i] = dp[i-1]$ o $dp[i] = dp[i-1] + 1$.

Si se desea saber la cantidad mínima de bombillos necesaria para poder alumbrar la calle desde el inicio hasta la posición $p_{k+1} - m - 1$, se puede lograr aplicando la desigualdad anterior, es decir:

$$dp[p_{k+1} - m - 1] = t \iff A[t-1] < p_{k+1} - m - 1 \leq A[t]$$

Entonces se puede afirmar que en cada iteración $k+1$ del algoritmo, con $k = 0 \ldots |postes|$, $p_{k+1} - m - 1$ cumple una de las siguientes opciones:

$$A[current - 1] < p_{k+1} - m - 1 \leq A[current]$$

$$A[current - 2] < p_{k+1} - m - 1 \leq A[current - 1]$$

Complejidad Temporal

Basándose en la idea del algoritmo se puede analizar su complejidad temporal según los siguientes pasos:

1. En las líneas 2-3 se ejecuta el método `check(n, postes, m)` que es $O(|postes|)$.

2. En las líneas 4-8 todas las operaciones son constantes respecto a la entrada.

3. Por último está el ciclo de la línea 9, que se ejecuta $|postes|$ veces y todas las operaciones contenidas son constantes respecto a la entrada.

Entonces la complejidad del algoritmo sería:

$$O(|postes|) + O(|postes|) = O(|postes|),$$

que por el principio de la suma es $O(|postes|)$.