El secreto de la Isla

Brian Ameht Inclán Quesada
Davier Sanchez Bello
Maykol Luis Martínez Rodríguez

Septiembre 2024

Contents

1 Generalización del problema

Eliminando las particularidades del problema (aldeas, guardianes, etc.) tenemos que: Dado un grafo $G = (V, E)$, debemos encontrar el subconjunto de vértices D más pequeño, tal que cada vértice de V pertenece a D o es adyacente a al menos un vértice de D.

Al problema anterior se le denomina problema del conjunto dominante (*dominating set problem*), un problema NP-completo.

2 Dominating Set Problem

Formalmente, el conjunto dominante se define como: Dado un grafo no dirigido $G = (V, E)$, un subconjunto de vértices $D \subseteq V$ se llama conjunto dominante si para cada vértice $u \in V \setminus D$ hay un vértice $v \in D$ tal que $(u, v) \in E$.

El número de dominancia de G se define como:

$$Y(G) := \min\{|S| : S \text{ es un conjunto dominante de } G\}$$

El *dominating set* en un grafo con vértices aislados, los incluye por fuerza. Luego, el problema de computar el *dominating set* de un grafo con vértices aislados puede ser resuelto computando el *dominating set* del mismo grafo sin los nodos aislados y luego incorporando estos al resultado. Por tanto, pasemos a demostrar que computar el *dominating set* de un grafo sin vértices aislados es NP-Hard. Para ello, parece natural realizar una reducción desde *Vertex Cover* hasta *Dominating Set*, puesto que cada cubrimiento de un grafo sin nodos aislados es un *dominating set* de este; aunque no necesariamente el *Minimum Dominating Set*.

3 Demostrando que es NP-Completo

Mostremos que el problema de decisión de decidir si existe un *dominating set* de tamaño k en un grafo es un problema NP-Completo.

El problema pertenece a NP, ya que una solución candidata puede ser verificada en tiempo polinomial, con tan solo computar el conjunto de vértices dominados y verificar que su tamaño sea igual al número total de vértices del grafo.

Ahora encontremos una reducción en tiempo polinomial desde el problema del *Vertex Cover* al problema del *Dominating Set*. Esto nos permitirá afirmar que *Dominating Set* es NP-Hard, con lo cual completaríamos la demostración de que es NP-Completo.

Para un grafo $G(V, E)$, definamos el grafo G', como el resultante de mantener todos los vértices y aristas de G y añadir un vértice w por cada arista $\langle u, v\rangle$. Tal vértice w tendrá solamente aristas a los vértices u y v. Sea D un *dominating set* de tamaño k de G'. Si $D \subset V$, entonces, D es un *vertex cover* de G, puesto que para cualquier nodo de V que no esté en D, este se encontrará a distancia 1 de un nodo en D. Es decir, para toda arista, al menos uno de sus extremos está en D. Si $D \not\subseteq V$, es posible obtener un *dominating set* D' de tamaño k en G' que sí cumpla $D \subset V$, tan solo con sustituir todo vértice w no perteneciente a V, con uno de los dos vértices a los que está conectado. Sea u tal vértice. u estará conectado a todos los nodos a los que lo estaba w, ya que w solo estaba conectado a u y a un nodo v tal que existe una arista entre u y v, por lo cual u también está conectado a él. Este conjunto D', sí será *vertex cover* de G.

Por otro lado, sea C un *vertex cover* de tamaño k del grafo G. Entonces, todo vértice de G se encuentra a una distancia de a lo más 1 de un vértice del *vertex cover*, ya que para cada arista al menos uno de sus extremos está incluido en el *vertex cover* y en este grafo no hay nodos aislados. Además, como para cada vértice w que no está en G pero sí en G', se cumple que tiene una arista hacia los dos nodos de una arista, y al menos uno de los nodos de la arista está en C, entonces w se encuentra a una distancia 1 de al menos un nodo en C.

El algoritmo de conversión es meramente añadir un vértice y dos aristas por cada arista original, lo cual es polinómico. Luego, como decidir si existe un *Vertex Cover* de tamaño k en un grafo G es equivalente a aplicar la conversión planteada, obteniendo el grafo G', y decidir si existe un *Dominating Set* de tamaño k en G'; entonces, el problema de *Dominating Set* es NP-Completo.

4 Encontrar la solución exacta

Para encontrar la solución exacta del problema del **Conjunto Dominante**, es necesario evaluar todas las combinaciones posibles de subconjuntos de vértices, lo cual puede tener una complejidad exponencial de $O(2^n)$, siendo n el número de vértices en el grafo. Sin embargo, existen algoritmos que mejoran esta complejidad.

4.1 Algoritmos que reducen la complejidad

Se han desarrollado algoritmos que mejoran el tiempo de resolución exacta utilizando técnicas avanzadas de **ramificación y poda**, logrando reducir la complejidad a $O(1.5^n)$. Estos algoritmos son ideales para grafos de tamaño moderado, permitiendo resolver el problema exacto en menos tiempo que los algoritmos ingenuos de fuerza bruta.

5 Algoritmos de Aproximación

Dado que encontrar la solución exacta es computacionalmente costoso para grafos grandes, los **algoritmos de aproximación** son una alternativa eficaz. Uno de los más comunes es el **algoritmo greedy**, que selecciona nodos de manera ávida, cubriendo en cada paso la mayor cantidad posible de nodos no cubiertos.

El **algoritmo greedy** no garantiza la solución óptima, pero ofrece una solución cercana en un tiempo razonable. La aproximación que ofrece está relacionada con el logaritmo del *grado máximo* (Δ) del grafo.

A continuación, se presenta la demostración de que el algoritmo greedy tiene una aproximación de $O(\ln(\Delta))$:

5.1 Aproximación del Algoritmo Greedy

Contexto del Problema

Estamos resolviendo el problema del **Conjunto Dominante** en un grafo, donde queremos seleccionar un subconjunto de nodos tal que:

- Cada nodo en el grafo esté cubierto directamente (pertenezca al conjunto) o tenga un vecino que esté en el conjunto.
- El objetivo es minimizar el tamaño de este conjunto dominante.

Idea del Algoritmo Greedy

El **Algoritmo Greedy** selecciona nodos de forma ávida, eligiendo siempre el nodo que cubre más nodos no cubiertos en cada paso, hasta que todos los nodos estén cubiertos. El objetivo es probar que este proceso nos da una solución que es, como máximo, $\ln(\Delta)$ veces más grande que la solución óptima.

Análisis Amortizado: Distribución del Costo

Cuando seleccionamos un nodo para el conjunto dominante, en lugar de asignar todo el costo de esa selección a ese nodo, distribuimos el costo entre todos los *nodos no cubiertos* que ese nodo acaba de cubrir.

Por ejemplo:

- Si seleccionamos un nodo v y este cubre a 5 nodos no cubiertos (incluyéndose a sí mismo si también estaba sin cubrir), asignamos una fracción del costo de $\frac{1}{5}$ a cada uno de esos 5 nodos.

Descomposición del grafo en estrellas

Supongamos que tenemos una solución óptima al problema, es decir, un conjunto dominante S^*. Sabemos que en S^*, cada nodo que no pertenece al conjunto dominante tiene un vecino que sí pertenece. Esto nos permite dividir el grafo en *"estrellas"*, donde:

- Cada estrella tiene un *nodo dominante* de S^* como "centro".
- Los *nodos no cubiertos* por otros están "conectados" a este centro.

El costo de la solución óptima es cubrir 1 estrella por cada nodo en S^*, lo que significa que el costo de cubrir todos los nodos de la estrella es 1.

Ejemplo de cómo se ve una estrella en un grafo:

```
  o        <- Nodo central (dominador)
 /|\
o o o     <- Nodos no cubiertos (vecinos del nodo central)
```

En este ejemplo:

- El nodo central o (dominador) está en el conjunto dominante S^*.
- Los nodos conectados debajo son *nodos no cubiertos* que dependen del nodo central para estar cubiertos en el grafo.

Costo amortizado del algoritmo greedy

En el *Algoritmo Greedy*, seleccionamos nodos basándonos en cuántos nodos no cubiertos pueden ser cubiertos en cada paso.

- Sea v el nodo dominante de una estrella en la solución óptima S^*.
- Sea $w(v)$ el número de nodos no cubiertos en la estrella de v.

Cuando el algoritmo greedy selecciona un nodo, cubre a $w(v)$ nodos, y asigna una fracción del costo de $\frac{1}{w(v)}$ a cada uno de los nodos cubiertos. Esto se hace porque si v puede cubrir muchos nodos, el costo se reparte entre todos ellos.

Una vez que un nodo ha sido cubierto, no recibe más costos.

Peor caso y costo total

En el peor caso, el algoritmo greedy selecciona nodos uno a uno, cubriendo un nodo en cada paso. El costo total que recibe cada nodo en una estrella es la suma de fracciones de costo, como:

$$\sum_{i=1}^{\delta(v)+1} \frac{1}{i} = \frac{1}{\delta(v)+1} + \frac{1}{\delta(v)} + \cdots + \frac{1}{2} + \frac{1}{1} = H(\delta(v)+1),$$

donde $H(\delta(v)+1)$ es la *función armónica*, que se comporta asintóticamente como:

$$H(\delta(v)+1) \approx \ln(\delta(v)+1) + O(1),$$

donde $\delta(v)$ es el número de vecinos del nodo v, y por tanto $H(\delta(v)+1)$ se aproxima a $\ln(\Delta)$, siendo Δ el grado máximo en el grafo.

El costo amortizado total por estrella en el peor caso es aproximadamente $\ln(\Delta)$. Esto significa que el número de nodos en el conjunto dominante calculado por el *Algoritmo Greedy* es como máximo $\ln(\Delta)$ veces mayor que el tamaño del conjunto dominante óptimo.

Por lo tanto, el algoritmo proporciona una aproximación de $\ln(\Delta)$ para el problema del conjunto dominante.

6 Resultados de los Tests

A continuación se muestran los gráficos obtenidos tras la ejecución de los tests que comparan las soluciones exactas y aproximadas, así como el análisis del logaritmo del grado máximo.

6.1 Comparación de Soluciones Exactas y Aproximadas

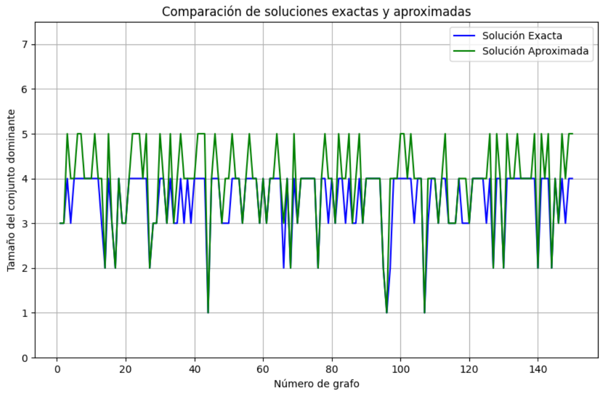

Figure 1: Comparación de Soluciones Exactas y Aproximadas

6.2 Logaritmo del Grado Máximo y Solución Aproximada

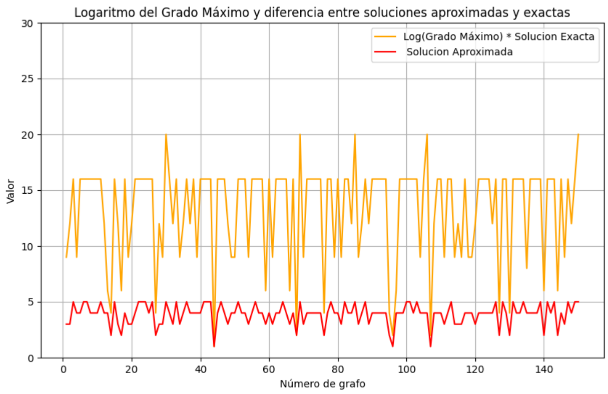

Figure 2: Logaritmo del Grado Máximo y Diferencia entre Soluciones

Los gráficos muestran claramente las diferencias entre las soluciones obtenidas de manera exacta y las soluciones aproximadas mediante el algoritmo greedy. El análisis del logaritmo del grado máximo también proporciona información sobre la calidad de las aproximaciones obtenidas.

References

[1] Dominating Set, Chapter 7. Disponible en: https://ac.informatik.uni-freiburg.de/teaching/ss_12/netalg/lectures/chapter7.pdf