Material de apoyo Teórica VII

Temario

Seguimos trabajando con problemas combinatorios:

- Problema de Satisfacibilidad booleana – SAT
- Uncapacitated Facility Location (UFL)
- Máximo conjunto independiente
- Problema de coloreo de grafos
- Problema de la mochila

Problemas combinatorios

- Son aquellos en los cuales se desea determinar combinaciones óptimas.
- Se caracterizan por tener un número finito de soluciones factibles.
- Generalmente este número es muy grande.

Satisfacibilidad booleana - SAT

Forma normal conjunctiva

$$(X1 \vee X2 \vee \overline{X3})$$
$$(\overline{X1} \vee \overline{X2} \vee X3)$$
$$(\overline{X1} \vee \overline{X2} \vee \overline{X3})$$
$$(\overline{X1} \vee X2 \vee X3)$$

Forma normal disyuntiva

$$(X1 \wedge X2 \wedge \overline{X3})$$
$$(\overline{X1} \wedge \overline{X2} \wedge X3)$$
$$(\overline{X1} \wedge \overline{X2} \wedge \overline{X3})$$
$$(\overline{X1} \wedge X2 \wedge X3)$$

SAT es el primer problema NP-Completo, lo demuestra Stephen Cook en 1971 (Universidad de Toronto), también lo hace de forma independiente Leonid Levin en 1973 (Academia de Ciencias de Rusia)

La demostración consiste en reducir una máquina de Turing no determinística a SAT

En 1972 Richard Karp presenta 21 problemas NP-Complete, en 1979 Garey y Johnson presentan más de 300 problemas NP-Complete

Encontrar si una formula en FNC es satisfacible es NP-Hard, si está escrita como FND es lineal, ¿cuánto cuesta pasar de FNC a FND?

Modelo matemático de la forma normal conjuntiva:

Se definen Xi como variables binarias o bivalentes (1 significa sí y 0 significa no)

Se define una variable bivalente Y que si vale 1 significa que el problema tiene solución y si vale cero significa que no la tiene.

Max Y

ST

X1 + X2 + (1 – X3) >= Y

(1 – X1) + (1 – X2) + X3 >= Y

(1 – X1) + (1 – X2) + (1 – X3) >= Y

(1 – X1) + X2 + X3 >= Y

Aplicaciones

Automatic Test-Pattern Generation, se generan test de entrada del circuito que al diferir la salida de un circuito testigo permiten detectar si una conexión en particular quedó “trabada” en un estado 0 o 1 o indicar que ese test no existe

Inteligencia artificial: se usa SAT para saber si se puede llegar a un objetivo en k pasos

Bioinformática: identificación de Haplotipos

Uncapacitated Facility Location (UFL)

Es el problema de distribución que vimos anteriormente (está en P) combinado con Set Covering (NP-Complete), UFL también es NP-Complete

Se debe decidir dónde abrir los depósitos y qué proporción de la demanda de los clientes satisface cada depósito abierto

Variables:

xij: Fracción de la demanda de la zona j (o del cliente j) que es satisfecha por el depósito ubicado en i.

yi: Variable binaria que cuando vale 1, significa que se establece un depósito en i

Constantes:

fi: Costo anual fijo de establecer un depósito en el lugar i

cij: Costo de producción y distribución si el depósito que está ubicado en i le proporcionara al cliente j todo lo que éste demanda.

El modelo matemático propuesto por Erlenkotter (1978) es el siguiente:

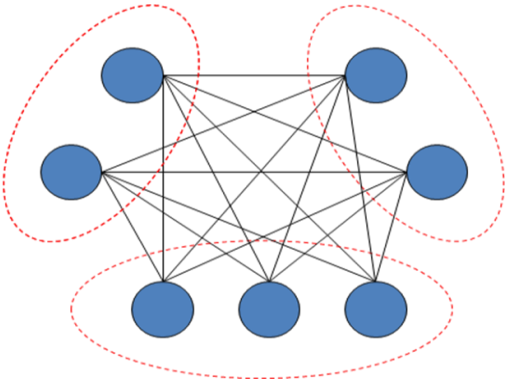
$$Minimizar \sum_i \sum_j c_{ij} x_{ij} + \sum_i f_i y_i$$

$$\text{s.t.} \sum_i x_{ij} = 1 \quad \text{for all } j$$

$$x_{ij} \leq y_i \quad \text{for all } i,\ j$$

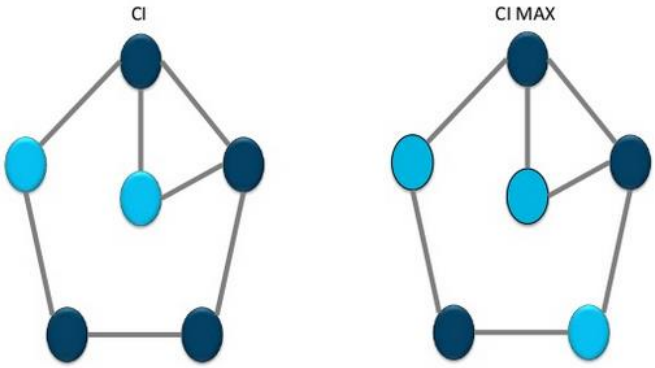
$$x_{ij} \geq 0, \quad y_i \in \{0,\ 1\} \quad \text{for all } i, j.$$

Máximo conjunto independiente



Conjunto independiente en un grafo es un conjunto de vértices tal que ninguno de los vértices es adyacente a otro

Conjunto independiente máximo es un conjunto tal que al agregarle un vértice más, deja de ser independiente



Su problema complementario (Clique en el grafo complemento) figura en los 21 problemas de Karp

Aplicaciones:

- Computer Vision/Pattern Recognition
- Information/Coding Theory
- Map Labeling
- Molecular Biology
- Scheduling

Problema de coloreo de grafos

Un Poco de Historia[1]

Hay problemas de la física, química, genética e informática, que pueden ser modelados como problemas en *grafos*. Sin embargo, este concepto tiene su origen en una anécdota pueblerina. Debemos remontarnos al siglo XVIII para encontrar la noción de *grafo* como un modelo matemático que permite analizar y resolver muchos problemas de la vida real.

La ciudad de Königsberg, Prusia, está atravesada por el río Pregel. En su curso, se encuentra una isla a partir de la cual el río se ramifica en dos. Existían 7 puentes que permitían pasar de una partea otra. Los habitantes de Königsberg se preguntaban si sería posible realizar un paseo atravesando cada uno de los puentes una única vez. Fallidos intentos hacían pensar que esto no era posible pero no se tenía certeza que así fuera. Fue en 1736 cuando el famoso matemático Leonhard Euler pudo asegurar que no era posible tal recorrido. El modelo matemático usado asociaba a cada zona de tierra firme un punto (*vértice*) y a cada puente una línea (*arista*) que unía los correspondientes puntos. El problema se tradujo en encontrar un recorrido por los *vértices* que pase una y sólo una vez por cada *arista*. Cualquier configuración de *vértices* y *aristas* es lo que conocemos hoy como *grafo*. Euler fue más allá del problema de los puentes de Könisberg y dio condiciones sobre la estructura de un *grafo* para que exista tal tipo de recorrido.

Tal vez el problema que más contribuyó al crecimiento de la teoría de *grafos* fue *el problema de los 4 colores*. En el año 1852, Francis Guthrie observó al colorear un mapa de Inglaterra que le bastaban 4 colores aún en el caso que dos regiones vecinas no pudieran ser coloreadas con el mismo color. Intrigado, escribe una carta a su hermano Frederick, discípulo del matemático De Morgan, planteándole la inquietud de si cualquier mapa dibujado en una hoja de papel tendría esta propiedad. Fredrick y De Morgan intentaron dar una respuesta matemática al problema. Pero, si bien no pudieron dibujar un mapa que requiriera al menos 5 colores, tampoco pudieron demostrar que 4 colores bastaban. El 23 de Octubre de 1852, De Morgan escribe una carta a Hamilton explicando el problema que termina diciendo:

If you return with some very simple case which makes me out a stupid animal, I think I must do as the sphynx did...

De Morgan no logró despertar el interés de Hamilton quien respondió que no disponía de tiempo para dedicarse al tema.

A fines del siglo XIX, la comunidad matemática comenzó a tratar el problema. En 1878 apareció la primera referencia escrita debida a Cayley en *The Royal Geographical Society*, quien también explicó ante la *London Mathematical Society* las dificultades encontradas hasta el momento para el tratamiento del problema.

En 1879, Alfred Kempe [52] publica en *American Journal of Mathematics* la primera demostración del problema que fue considerada correcta. Sin embargo,

en 1890 Percy John Heawod [44] encuentra un error en los argumentos usados por Kempe y contribuye con el resultado que 5 colores son suficientes para colorear un mapa.

Durante muchos años diversos investigadores abordaron el tema. Hubo infructuosos intentos de demostración, pero muchos de ellos aportaron resultados parciales o formulaciones equivalentes que enriquecieron la teoría. Recién en 1976, Wolfgang Haken y Kenneth Appel [2, 3] logran demostrarla conjetura con la ayuda de una computadora. Su demostración está basada en conceptos que ya habían sido utilizados por Kempe, Heawod y Heesch en sus fallidas demostraciones: reductibilidad y descarte. Esencialmente, si la conjetura fuera falsa ninguna configuración entre 1478 posibles podría aparecer en el mapa contraejemplo (reductibilidad). Sin embargo, se demuestra que el mapa contiene alguna de las 1478 configuraciones (descarte) llegando a una contradicción. El procedimiento de reductibilidad y descarte de los 1478 casos fue efectuado con una computadora. Esta metodología no fue aceptada universalmente como una demostración ante la dificultad de verificar la validez del razonamiento de la computadora. En 1997, Robertson, Sanders, Seymour y Thomas [69] logran reducir a 633 las configuraciones a tener en cuenta y utilizan un procedimiento computacional más transparente que es aceptado como una demostración formal de la conjetura.

De la misma manera que el problema de los puentes de Könisberg, el *problema de los 4 colores* puede ser modelado como un problema de *grafos*. Basta asignar un *vértice* a cada país y una *arista* entre *vértices* que correspondan a países fronterizos.

Pero la historia no termina con la coloración de un mapa. Además de la conjetura de los 4 colores que despertó durante muchos años el interés de los investigadores, paralelamente se desarrolló el área de teoría de *grafos*. Entre los muchos problemas estudiados surgió una natural extensión del problema de los 4 colores: el coloreo de un *grafo*. Un *coloreo* de un *grafo* es cualquier asignación decolores a los *vértices* de tal manera que *vértices* conectados por una *arista* no comparten el mismo color. Es inmediato plantearse la pregunta: ¿cuál es la cantidad mínima de colores necesarios para colorear un *grafo* y cómo colorear el *grafo* con esa cantidad de colores? Este es el conocido *problema de coloreo de grafos*.

No se ha encontrado aún una respuesta teórica a esta pregunta para cualquier configuración de un *grafo*. Los resultados teóricos de la literatura podemos encuadrarlos en dos tipos. Por un lado, aquellos que han caracterizado familias de *grafos* para las cuales se puede establecer la cantidad mínima de colores y, en algunos casos, cómo colorear con esa cantidad de colores. Por otro lado, resultados que brindan cotas superiores e inferiores para *grafos* generales o con algunas propiedades particulares.

Más allá del interés teórico que puedan tener los problemas en *grafos*, muchos de ellos y en particular el *problema de coloreo de grafos* tienen una significativa importancia práctica.

Aplicaciones del *Problema de Coloreo de Grafos*[1]

Son numerosas las situaciones de la vida real en las cuales surgen problemas que pueden ser modelados como un *problema de coloreo de grafos*. A continuación mencionaremos algunas.

Asignación de horarios

Supongamos que queremos asignar turnos para el dictado de un conjunto de materias. Claramente deberá considerarse que dos materias dictadas por el mismo profesor no podrán ser asignadas a turnos que se superpongan. El mismo argumento es aplicable a materias dictadas a un mismo grupo de estudiantes. El problema de determinar el mínimo número de turnos para dictar todas las materias es equivalente al *problema de coloreo* en el *grafo* que resulta de considerar un *vértice* por cada posible materia-turno y una *arista* por cada incompatibilidad horaria. Asociando los turnos a colores, es inmediata la equivalencia de los problemas. Este problema se encuadra dentro de una clase más general que es el de programación de actividades que deban ser ejecutadas por máquinas y/o personas. El objetivo es encontrar la cantidad mínima de máquinas y/o personas necesarias para realizar un conjunto de tareas. El caso más sencillo es cuando cualquier tarea puede ser realizada por cualquier máquina/persona y existen ciertas incompatibilidades entre las tareas que impiden que éstas sean realizadas en la misma máquina. Por ejemplo, si dos tareas deben ser realizadas en el mismo horario.

Asignación de Frecuencias Radiales

Consideremos un conjunto de usuarios (radios o particulares) del espectro electromagnético a los cuales hay que asignar frecuencias. Para evitar interferencias, las zonas geográficamente cercanas deberán tener distintas frecuencias mientras que aquellas que se encuentren suficientemente lejos podrán compartirlas. Determinar cómo asignar el mínimo número de frecuencias necesarias para suplir los requerimientos de todos es equivalente al *problema de coloreo de grafos*. Basta considerar un *vértice* por cada usuario y una *arista* entre dos *vértices* cuando la distancia geográfica entre los usuarios es inferior a una distancia establecida para evitar interferencias. Si cada frecuencia es identificada con un color, encontrar la mínima cantidad de colores para colorear el *grafo* resuelve el problema. En la práctica pueden surgir algunos otros requerimientos y/u objetivos. Por ejemplo, los usuarios pueden tener restricciones sobre el uso de las frecuencias. Por otro lado, ante una situación en la que sea imposible evitar interferencias, es de utilidad saber cuál es aquella asignación que minimiza la probabilidad global de que se produzca una interferencia.

Asignación de Registros

Durante la ejecución de un programa, las variables que se encuentran en los registros de memoria pueden ser accedidas más rápidamente que aquellas que no

se encuentran en los mismos. En la mayoría de los casos, la cantidad de variables supera el número de registros disponibles, por lo cual un registro deberá ser usado en distintos tiempos de ejecución por diferentes variables. Con el objetivo de minimizar el tiempo de ejecución, es fundamental que aquellas variables a las cuales deba accederse dentro de un mismo período corto de tiempo estén accesibles en los registros de memoria. Es decir, es necesario minimizar los accesos a memoria fuera de los registros. Se puede crear un *grafo* con un *vértice* por cada variable del programa y *aristas* entre *vértices* que correspondan a variables necesarias en un mismo período de tiempo. Si identificamos cada registro con un color, cualquier coloreo del *grafo* brinda una posible relación variable-registro que no asigna un mismo registro a variables requeridas en igual período. Si el mínimo número de colores necesario para colorear el *grafo* es inferior a la cantidad de registros disponibles, se optimiza el uso de la memoria. Debido al interés práctico de esta aplicación, hay numerosos trabajos así como variaciones del problema.

[1] Material extraído de la Tesis Doctoral: Problema de coloreo de grafos “Un estudio poliedral para un algoritmo Branch-and-cut” Isabel Méndez Díaz, Departamento de Computación, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires

Modelización

Modelo Clásico

$$Minimizar \quad \sum_{j=1}^{n} w_j$$

$$sujeto \quad a$$

$$\sum_{j=1}^{n} x_{ij} = 1 \qquad \forall\, i \in V$$

$$x_{ij} + x_{kj} \leq w_j \qquad \text{si } [i,k] \in E \;\; \forall\, j = 1, \ldots, n$$

$$x_{ij}, w_j \in \{0,1\} \qquad \forall\, i \in V \;\; \forall\, j = 1, \ldots, n$$

Xij = 1 ➔🡐 el vértice i se colorea con color j

Funcional: El modelo clásico minimiza la cantidad de colores.

Sumatoria igualada a 1: Cada vértice debe estar coloreado con exactamente un color

Restricción:

Si dos vértices son adyacentes no pueden tener el mismo color

Si algún vértice usa el color j se fuerza wj a valer 1 (notar que no se fuerza el cero, esto agrega soluciones no óptimas)

Modelo por conjuntos independientes

Colorear es determinar a qué conjunto independiente pertenece cada vértice, cada conjunto luego se colorea con un mismo color.

Sea $S = \{S_1, S_2, \ldots, S_t\}$ el conjunto de todos los conjuntos independientes de G. Sea x_{S_i} una variable binaria tal que $x_{S_i} = 1$ si todos los *vértices* de S_i son coloreados con un mismo color y $x_{S_i} = 0$ en caso contrario. Con estas variables el problema de coloreo puede ser formulado como:

$$
\begin{aligned}
&Minimizar \quad \sum_{i=1}^{t} x_{S_i} \\
&sujeto \quad a \\
&\qquad \sum_{i:v \in S_i} x_{S_i} = 1 \qquad \forall \ v \in V \\
&\qquad\qquad x_{S_i} \in \{0,1\} \qquad \forall\, i = 1, \ldots, t
\end{aligned}
$$

Muchísimas variables, enumerar los conjuntos independientes toma mucho tiempo

Funcional: utilizar la menor cantidad de conjuntos independientes

Sumatoria: Cada vértice debe pertenecer a un conjunto independiente

Modelo por conjuntos independientes maximales

Para reducir la cantidad de variables solo se modelan los conjuntos independientes maximales (conjuntos independientes a los que no se les puede agregar ningún vértice y que sigan siendo independientes), se permite que un vértice pertenezca a más de un conjunto (después es arbitrario el color de cual se le asigna).

Si $M = \{M_1, M_2, \ldots, M_p\}$ es el conjunto de todos los conjuntos independientes maximales, la formulación es:

$$
\begin{aligned}
&Minimizar \quad \sum_{i=1}^{p} x_{M_i} \\
&sujeto \quad a \\
&\qquad \sum_{i:v \in S_i} x_{M_i} \geq 1 \qquad \forall \ v \in V \\
&\qquad\qquad x_{M_i} \in \{0,1\} \qquad \forall\, i = 1, \ldots, p
\end{aligned}
$$

Funcional: utilizar la menor cantidad de conjuntos independientes maximales

Sumatoria: Cada vértice debe pertenecer a un conjunto independiente o más

Este modelo tiene menos variables que el de conjuntos independientes, igual es muy grande, enumerar los conjuntos maximales también toma mucho tiempo (no es polinomial).

Se reduce la cantidad de variables usando solo los conjuntos independiente maximales (no se les puede agregar ningún vértice y que sigan siendo conjuntos independientes), se permite que un vértice pertenezca a más de un conjunto (después es arbitrario el color de cual se le asigna)

Un poco menos de variables que el anterior, igual es MUY grande y enumerar los conjuntos maximales no es polinomial (toma mucho tiempo)

Resolución

El problema de coloreo de grafos pertenece a la categoría de problemas NP-Difícil, por esta razón es importante distinguir si buscamos una solución exacta (algo que en muchos casos no podremos lograr) o una aproximada.

Los mejores resultados exactos hasta el momento se consiguen utilizando programación lineal entera y aplicando un algoritmo de Branch&Cut.

Si el tamaño o las características particulares del problema no permiten encontrar una solución exacta con los algoritmos conocidos (el tiempo de corrida es inmanejable) se debe recurrir a heurísticas para encontrar una solución aproximada.

Resolución de la relajación lineal

Resolver la relajación lineal de un problema con variables enteras implica resolverlo como si todas las variables fueran continuas (se relaja la restricción de que sean enteras).

¿qué pasa cuando resolvemos la relajación lineal del problema de coloreo de grafos?

$$Minimizar \quad \sum_{j=1}^{n} w_j$$

$$sujeto \quad a$$

$$\sum_{j=1}^{n} x_{ij} = 1 \qquad \forall\, i \in V$$

$$x_{ij} + x_{kj} \leq w_j \qquad \text{si } [i,k] \in E \;\; \forall\, j = 1, \ldots, n$$

$$x_{ij}, w_j \in \{0,1\} \qquad \forall\, i \in V \;\; \forall\, j = 1, \ldots, n$$

Observen que hemos tachado la restricción que indica que las variables deben tomar valor entero. Resolver la relajación lineal de un problema implica tachar esa restricción.

¿Cuál es la solución óptima?

Si las variables son continuas el Z es como mucho 2, ya que la solución Xi1 = 0,5 y Xi2 = 0,5 para todos los vértices es factible, por esta razón la formulación es débil (esto tiene sus contras pero no la descarta ya que es mucho más compacta y fácil de generar que otras más fuertes)

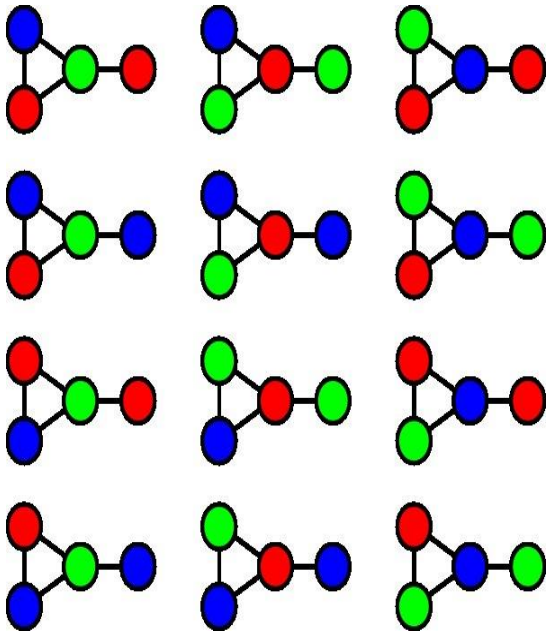
Otro de los problemas en la solución es la SIMETRÍA.

Simetría

Cuando se resuelve de manera exacta un problema de coloreo de grafos aparecen muchas soluciones alternativas.

Por ejemplo, si un grafo de cuatro vértices se puede pintar con tres colores, al resolverlo de manera exacta, encontrará todas las soluciones distintas que tienen

3 colores, pero no nos sirven todas las soluciones que tienen igual valor de función objetivo, nos basta una. **Esto se llama problema de simetría**.



¿Cómo hacemos para que el modelo no encuentre todas las soluciones que tienen la cantidad óptima de colores?

Eliminación de simetría

$$w_j \leq \sum_{i \in V} x_{ij} \qquad \forall\, j = 1, \ldots, n$$

$$w_j \geq w_{j+1} \qquad \forall\, j = 1, \ldots, n-1$$

La primera restricción sirve para que “Si ningún vértice tiene un color no permitirle tomar valor (esto es forzar el cero)”.

La segunda restricción sirve para “No permitir “saltear” colores (forzar un coloreo ordenado)”

Podemos agregar una restricción que agrupe las dos condiciones anteriores.

$$\sum_{i=1}^{n} x_{ij} \geq \sum_{i=1}^{n} x_{ij+1} \qquad \forall\, j = 1, \ldots, n-1$$

Esto se debe a que la restricción ordena los colores en función de la cantidad de vértices que colorean, esto incluye no saltear colores y aparte quita otras soluciones.

¿Qué nos queda de esta clase?

- Seguimos trabajando con problemas combinatorios que vamos a encontrar en la Guía 3 (enteras)
  - Satisfacibilidad booleana (SAT)
  - Mochila / Knapsack
  - Uncapacitated facility location (UFL)
  - Coloreo de vértices en grafos

Y ya pueden resolver toda la Guía 3