Tarea 6
Informática Teórica

Matías Peñaloza
202373037-8

2024-2

Concepto	**Tiempo [min]**
Revisión	30
Desarrollo	90
Informe	30

1 Enunciado

Una *partición* de un conjunto $\mathscr{A}$ es un colección de subconjuntos de $\mathscr{A}$ que son disjuntos a pares y cuya unión es $\mathscr{A}$.

El problema 1-IN-3 SAT da una fórmula lógica ϕ en 3CNF, pregunta si hay una asignación de variables que satisfacen ϕ tal que cada cláusula contiene exactamente un literal verdadero y dos falsos. Demostramos en clases que 1-IN-3 SAT es NP-completo.

El problema CONTAINS PARTITION pone un conjunto $\mathscr{U}$ y una colección $\mathscr{C}$ de subconjuntos de $\mathscr{U}$. Pregunta si $\mathscr{C}$ contiene una partición de $\mathscr{U}$. Demuestre que CONTAINS PARTITION es NP-completo, usando el problema 1-IN-3 SAT visto en clase.

2 Desarrollo

Para que CONTAINS PARTITION sea NP-completo, debe cumplir la condición de ser NP y que todos los problemas NP se reduzcan a este (NP-hard).

2.1 CONTAINS PARTITION en NP

Para demostrar que CONTAINS PARTITION está en NP, basta con dar un certificado que verifique que la solución dada satisface el problema en tiempo polinomial. En este caso, dados un conjunto $\mathscr{U}$ y una colección $\mathscr{C}$ de subconjuntos, se puede comprobar que los subconjuntos de $\mathscr{C}$ son disjuntos a pares y que la unión de estos es $\mathscr{U}$ utilizando operaciones entre conjuntos como intersección y unión, las cuales, para conjuntos finitos, pueden ser computadas en tiempo polinomial y siempre entregar una respuesta. Por lo tanto, podemos decir que CONTAINS PARTITION está en NP.

2.2 CONTAINS PARTITION es NP-hard

Para demostrar que CONTAINS PARTITION es NP-hard, basta con reducir polinomialmente un problema NP a CONTAINS PARTITION, que en este caso será 1-IN-3 SAT, por lo que procederemos a la demostración.

Partiremos de la ecuación 1-IN-3 SAT de la forma:

$$\phi(x_1, x_2, ..., x_n) \equiv R_1 \wedge ... \wedge R_i \wedge ... \wedge R_m$$

donde cada R_i es una cláusula que contiene 3 literales y que solo es verdadera si uno de los 3 literales contenidos en ella es verdadero. Modificaremos esta ecuación para que ninguna aparición de x_1 o x_2 esté negada, por lo tanto, cada $\bar{x}_1$ o $\bar{x}_2$ se negará otra vez para obtener $\bar{\bar{x}}_1 \equiv x_1$. Además, agregaremos $n-2$ cláusulas de manera que todas las $x_i, i \neq 1, 2$, sean falsas:

$$\phi^*(x_1, x_2, ..., x_n) \equiv \phi(x_1, x_2, ..., x_n) \wedge (\bar{x}_3 \vee a \vee b) \wedge (\bar{x}_4 \vee a \vee b) ... \wedge (\bar{x}_n \vee a \vee b)$$

donde a y b son siempre falsas. De este modo, obtenemos una ecuación 1-IN-3 SAT que solo es verdadera si x_1 y x_2 son verdaderos y además cada cláusula R_i contiene a x_1 o a x_2, pero no ambos.

Lo que se trata de hacer aquí es que cada cláusula R_i represente un ítem I_i del conjunto $\mathscr{U}$, y que cada aparición de un x_1 o x_2 represente que ese ítem pertenece al subconjunto $\mathscr{U}_1$ o $\mathscr{U}_2$ dentro de $\mathscr{C}$. De esta manera, la ecuación ϕ^* solo es verdadera si:

1. Cada ítem I_i de $\mathscr{U}$ pertenece a $\mathscr{U}_1$ o $\mathscr{U}_2$. (Cada R_i debe ser verdadero conteniendo a x_1 o x_2).
2. Los conjuntos $\mathscr{U}_1$ y $\mathscr{U}_2$ no repiten ítems entre sí. (No pueden existir x_1 y x_2 en un mismo R_i).

Esto traduce cada instancia de 1-IN-3 SAT, donde tenemos una ecuación ϕ transformada en ϕ^* y una combinación de variables $(x_1, ..., x_n)$, en la instancia de CONTAINS PARTITION donde se tiene un conjunto $\mathscr{U}$ y una colección $\mathscr{C}$ que contiene dos subconjuntos $\mathscr{U}_1$ y $\mathscr{U}_2$ de $\mathscr{U}$. Aquí, ϕ^* solo es verdadera si $\mathscr{C}$ contiene la partición formada por $\mathscr{U}_1$ y $\mathscr{U}_2$.

El algoritmo que ejecuta la transformación de ϕ a ϕ^* está acotado por el polinomio:

$$3m + 3 * (n-2) \leq 3m + 3n$$

donde $3m$ representa la búsqueda y transformación de las apariciones x_1 o x_2 a lo largo de las m cláusulas, las cuales contienen 3 variables, y $3 * (n-2)$ representa agregar una cláusula que contiene 3 variables por cada una de las $n-2$ variables restantes de ϕ.

Ya que logramos traducir todas las instancias de 1-IN-3 SAT (un problema que está en NP) a CONTAINS PARTITION con un algoritmo acotado por un polinomio y que siempre entrega una respuesta (1-IN-3 SAT $\leq_p$ CONTAINS PARTITION), concluimos que CONTAINS PARTITION es NP-hard.

Nota*: Cabe resaltar que siempre $n > 2$ y $m \geq 1$ ya que 1-IN-3 SAT requiere de por lo menos 3 variables y 1 cláusula, por lo que tiene sentido reducir 1-IN-3 SAT a CONTAINS PARTITION, que necesita de $m \geq 1$ ítems en $\mathscr{U}$.

2.3 CONTAINS PARTITION es NP-completo

Dado que demostramos que CONTAINS PARTITION está en NP y en NP-hard, podemos decir que CONTAINS PARTITION es NP-completo.