Teoría de la NP-completitud

Luz Marina Moreno de Antonio y Jorge Riera Ledesma

Departamento de Ingeniería Informática y de Sistemas. Universidad de La Laguna

17 de noviembre de 2016

Capítulo 1

Teorema de Cook

1.1. Problema involucrado

SATISFACTIBILIDAD (SAT)

ENTRADA: Un conjunto de cláusulas $C = \{c_1, c_2, \ldots, c_m\}$ sobre un conjunto finito U de variables.
PREGUNTA: ¿Existe una asignación booleana para U, tal que satisfaga todas las cláusulas de C?

Teorema 1. *SAT es $\mathcal{NP}$-completo.*

Demostración. Es fácil comprobar que SAT $\in \mathcal{NP}$, ya que se puede encontrar una algoritmo para una Máquina de Turing No Determinista (NDTM) que reconozca el lenguaje $L(3\text{SAT}, e)$, para un esquema de codificación e, en un número de pasos acotado por una función polinomial. De esta manera, el primer requerimiento para la prueba de $\mathcal{NP}$-completitud se cumple.

Para el segundo requerimiento debemos bajar a nivel de lenguaje, donde SAT se representa por un lenguaje $L_{SAT} = L[SAT, e]$ para algún esquema de codificación razonable e. Debemos demostrar que para todo lenguaje $L \in \mathcal{NP}$, $L \preceq L_{SAT}$. Los lenguajes de la clase $\mathcal{NP}$ son bastante diversos, se trata de una clase infinitamente grande, por lo que no esperemos presentar una demostración para cada uno de los lenguajes por separado.

No obstante, todo lenguaje de la clase $\mathcal{NP}$ puede describirse de una manera estándar: mediante un programa para una NDTM que lo reconozca en tiempo polinomial. Esto nos permitirá trabajar con un programa genérico para una NDTM y derivar una transformación polinomial genérica desde el lenguaje que reconozca esta máquina genérica al lenguaje L_{SAT}. Esta transformación genérica, cuando se particularice para un programa de una NDTM M que reconozca el lenguaje L_M, dará la deseada transformación desde L_M a L_{SAT}. Así, en esencia, presentaremos una demostración simultánea para todos los lenguajes $L \in \mathcal{NP}$ de que $L \preceq L_{SAT}$.

Para empezar, denotemos por M un programa arbitrario en tiempo polinomial para una NDTM, especificado por Γ, Σ, b, Q, q_0, q_Y, q_N, y δ, que reconoce el lenguaje $L = L_M$. Además, sea $p(n)$ un polinomio que acota superiormente la función $T_M(n)$. Sin pérdida de generalidad se asume que $p(n) \geq n$ para todo $n \in \mathbb{Z}^+$. La transformación genérica f_L será derivada en términos de M, Γ, Σ, b, Q, q_0, q_Y, q_N, δ y p.

Por razones de conveniencia describiremos f_L como si fuera una función entre el conjunto de las cadenas de Σ hasta las entradas del problema SAT, mas que hasta las cadenas del alfabeto inducido por el esquema de codificación asociado a SAT, ya que los detalles de la demostración asociados al esquema de codificación pueden ser deducidos fácilmente.

De esta manera, f_L tendrá la propiedad de que para todo $x \in \Sigma^*$, $x \in L$ si, y sólo si, $f_L(x)$ es satisfecho por una asignación booleana. La clave de las construcción de f_L está en demostrar cómo un conjunto de cláusulas puede ser usado para comprobar si una entrada x es aceptada por el programa para una NDTM M, es decir, si $x \in L$.

Si la cadena $x \in \Sigma^*$ es aceptada por M, entonces, habrá una secuencia de estados de M que acepte x acotada superiormente por un polinomio $p(n)$, donde $n = |x|$. Dicha secuencia no puede involucrar ninguna celda de la cinta excepto aquellas comprendidas entre $-p(n)$ y $p(n) + 1$ ya que la la cabeza de lectura/escritura comienza en la celda 1, y se mueve una única posición en cada paso. El estado de la computación en cualquier instante puede determinado completamente dando el contenido de las celdas, el estado actual, y la posición de la cabeza de lectura/escritura. Es más, puesto que no se llevarán a cabo más de $p(n)$ pasos durante la ejecución, habrá a los sumo $p(n) + 1$ instantes que deben ser considerados. Esto nos permitirá describir la computación completamente usando sólo un número limitado de variables booleanas y una asignación booleana para las mismas.

A continuación se muestra cómo f_L construye el conjunto de variables a partir de M. Se etiquetarán los estados de Q como $q_0, q_1 = q_Y, q_2 = q_N, q_3, \ldots, q_r$, donde $r = |Q| - 1$, y los elementos de Γ como $s_0 = b, s_1, s_2, \ldots, s_v$, donde $v = |\Gamma| - 1$. Se crearán tres tipos de variables, cuyo sentido se especifica en la tabla 1.1.

Tabla 1.1: Variables creadas por f_l

Variable	*Rango*	*Significado*
$Q[i,k]$	$0 \leq i \leq p(n)$ $0 \leq k \leq t$	En el instante i, M está en el estado q_k
$H[i,i]$	$0 \leq i \leq p(n)$ $-p(n) \leq j \leq p(n)+1$	En el instante i, la cabeza de lectura/escritura está examinando la celda j
$S[i,j,k]$	$0 \leq i \leq p(n)$ $-p(n) \leq j \leq p(n)+1$ $0 \leq k \leq v$	En el instante i, el contenido de la celda j es el símbolo s_k

□

Siglas

NDTM Máquina de Turing No Determinista. 3, 4

SAT SATISFACTIBILIDAD. 3