Teorema de Cook-Levin

Adrián Rodríguez Bazaga, Eleazar Díaz Delgado

Rudolf Cicko

Complejidad Computacional, Grado en Ingeniería Informática
Universidad de La Laguna
Curso 2016-2017

Contenidos

Preámbulo

Teorema de Cook: ¿Qué es?

Problema de la satisfactibilidad (SAT)

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?

SAT es $\mathcal{NP}$-completo I

Primero se demuestra que SAT está en $\mathcal{NP}$. Para ello se puede generar una máquina de Turing no determinista(NDTM), que resuelva SAT en tiempo polinomial. Es decir, que dada una fórmula booleana ϕ se hallen los valores de entrada, tal que se satisfaga ϕ usando dicha NDTM.

SAT es $\mathcal{NP}$-completo II

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.

SAT es $\mathcal{NP}$-completo III

Denotemos por M un programa arbitrario en tiempo polinomial para una Máquina de Turing No Determinista (NDTM), especificado por Γ, Σ, b, Q, q_0, q_Y, q_N, y δ. Que decide un lenguaje A en un tiempo polinómico n^k.

Se representará la M sobre un tablero, en el cual cada fila es representa una configuración:

- La primera fila representa la configuración inicial de la máquina.
- Se acepta el lenguaje si+ alguna fila del tablero es verdadera.

SAT es $\mathcal{NP}$-completo: tabla

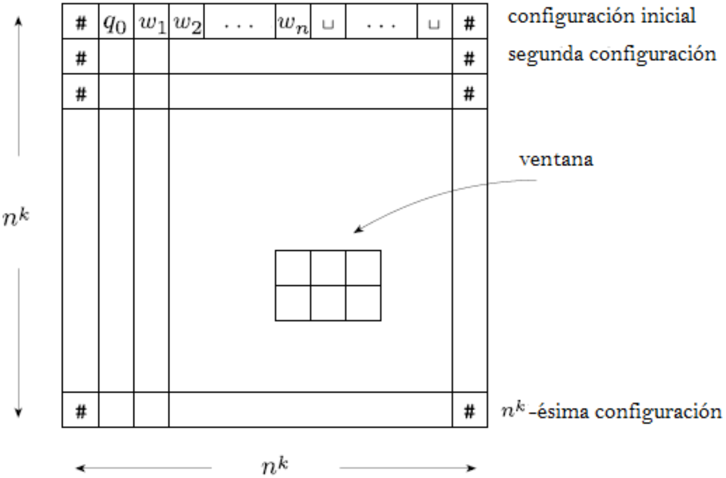

Figura: Tabla de configuraciones

SAT es $\mathcal{NP}$-completo IV

Generación de la fórmula ϕ

$$\phi = \phi_{cell} \wedge \phi_{start} \wedge \phi_{move} \wedge \phi_{accept}$$

SAT es $\mathcal{NP}$-completo V

Generación de la fórmula ϕ_{cell}

$$\phi_{cell} = \bigwedge_{1 \leq i,j \leq n^k} \left[\left(\bigwedge_{s \in C} x_{i,j,s} \right) \wedge \left(\bigwedge_{s,t \in C \wedge s \neq t} (\overline{x_{i,j,s}} \vee \overline{x_{i,j,t}}) \right) \right]$$

SAT es $\mathcal{NP}$-completo VI

Generación de la fórmula ϕ_{start}

$$
\begin{aligned}
\phi_{start} = {} & x_{1,1,\#} \wedge x_{1,2,q_0} \wedge \\
& x_{1,3,w_1} \wedge x_{1,4,w_2} \wedge \cdots \wedge x_{1,n+2,w_n} \wedge \\
& x_{1,n+3,\sqcup} \wedge \cdots \wedge x_{1,n^k-1,\sqcup} \wedge x_{1,n^k,\#}
\end{aligned} \tag{1}
$$

SAT es $\mathcal{NP}$-completo VII

Generación de la fórmula ϕ_{accept}

$$\phi_{accept} = \bigvee_{1 \leq i,j \leq n^k} x_{i,j,q_{accept}}$$

SAT es $\mathcal{NP}$-completo VIII

Generación de la fórmula ϕ_{move}

$$\phi_{move} = \bigwedge_{1 \le i,j \le n^k} (ventana\ en\ i,j\ que\ es\ legal)$$

Generación de la fórmula de la ventana

$$\bigvee_{a_1 \ldots a_6} x_{i,j-1,a_1} \wedge x_{i,j,a_2} \wedge x_{i,j+1,a_3} \wedge x_{i+1,j-1,a_4} \wedge x_{i+1,j,a_5} \wedge x_{i+1,j+1,a_6}$$

Conclusión: SAT es $\mathcal{NP}$-completo

- Para todo problema $L \epsilon NP$, existe un f_L que es una transformación polinomial de L al SAT

Referencias I

Michael Sipser
Introduction to the Theory of Computation.
Third edition, Cengage Learning, 2013

Michael R. Garey, David S. Johnson
Computers and Intractability: A Guide to the Theory of NP-Completeness.
W.H. Freeman. ISBN 0-7167-1045-5.

Cook, Stephen
The complexity of theorem proving procedures.
Proceedings of t he Third Annual ACM Symposium on Theory of Computing. pp. 151-158 (1971)

Karp, Richard M.
Reducibility Among Combinatorial Problems.
Complexity of Computer Computations. New York: Plenum, pp. 85-103. ISBN 0-306-30707-3 (1972)

Referencias II

T. P. Baker, J. Grill, R. Solovay
Relativizations of the P=NP question.
SIAM Journal on Computing. 4 (4): 431-442 (1975)

Dekhtiar, M.
On the impossibility of eliminating exhaustive search in computing a function relative to its graph.
Proceedings of the USSR Academy of Science. 14: 1146-1148.

Levin, Leonid
Universal search problems.
Problems of Information Transmission. 9 (3): 115-116 (1973)

Design and Analysis of Algorithms
Lecture, School of Information and Computer Sciences, University of California, Irvine
https://www.ics.uci.edu/~eppstein/161/960312.html

Referencias III

NP-complete problems
Lecture, Electrical Engineering and Computer Sciences, Berkeley
https://people.eecs.berkeley.edu/~vazirani/algorithms/chap8.pdf