Decidibilidad y complejidad

Curso 2018-2019

UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

Senén Barro Ameneiro, CiTIUS

@SenenBarro

Material elaborado fundamentalmente por
el profesor Manuel Mucientes Molina

Bibliografía

- J.E. Hopcroft, R. Motwani y J.D. Ullman, "Teoría de Autómatas, Lenguajes y Computación", Addison Wesley, 2008.
  - Capítulo 9 y 10

- P. Linz, ''An Introduction to Formal Languages and Automata'', Jones and Bartlett Publishers, Inc., 2001.
  - Capítulos 11, 12 y 14

Lenguajes recursivos y recursivamente enumerables

Un lenguaje L es **recursivamente enumerable** (LRE) si existe una MT que acepta cualquier cadena del lenguaje y se para:

- Si $w \in L$, $q_0w \mid\!-^* x_1q_fx_2$, $q_f \in F$

Sin embargo:

- Si $w \notin L$, la MT **se parará** en un estado no final –rechazándola- o **entrará en un bucle infinito**

Un lenguaje L sobre un alfabeto Σ es **recursivo** (LRC) si existe una MT que acepta L y se para con cualquier cadena $w \in \Sigma^+$ **-se para la acepte o no**-

Hay lenguajes que no son LRE. La demostración es compleja, puesto que todos los lenguajes descritos de forma algorítmica son aceptados por MT (son por tanto LRE)

Computabilidad y decibilidad

- Una función *f* es computable en un dominio si existe una MT que computa el valor de *f* **para todos los argumentos del dominio**

- Si el resultado de la computación de un problema es si/no, se habla de decidibilidad/indecidibilidad

- Problemas decidibles: existe una MT que da la respuesta correcta (si/no) para cada argumento del dominio

Problema de la parada en MT

Sea w_M una cadena que describe la MT M, y sea w una cadena sobre el alfabeto de M. Una solución al problema de la parada sería una MT H en la que, para cualesquiera w_M y w, se ejecuta la computación:

- $q_0 w_M w \vdash^* x_1 q_y x_2$, si M aplicada a w se para
- $q_0 w_M w \vdash^* x_1 q_n x_2$, si M aplicada a w no se para
- q_y y q_n son estados finales de H

- No existe ninguna MT H que se comporte como se requiere para el problema de la parada: problema indecidible
- Si el problema de la parada fuese decidible, entonces todos los LRE serían LRC

Complejidad computacional

- <u>Ejemplo</u>: dado un vector de enteros, ¿cuánto tiempo se tardará en ordenarlo?

- Para determinar la complejidad
  - Usaremos MT
  - El tamaño del problema será n
  - Interesa saber cuánto aumenta el tiempo al incrementar n

- Si una computación tiene complejidad (de tiempo) T(n) significa que puede ser resuelta en no más de T(n) movimientos de una MT para un tamaño problema de n

- No nos interesará el tiempo exacto, pero sí el orden de magnitud: O(…)

MT y complejidad

Desde el punto de vista de la decidibilidad, todas las MT son equivalentes. Desde el punto de vista de la complejidad no

<u>Ejemplo</u>: $L = \{a^n b^n : n \geq 1\}$

- MT estándar: $O(n^2)$
- MT con dos cintas: $O(n)$

Problema de la satisfacibilidad (SAT)

- Expresiones en forma normal conjuntiva: $e = t_i \wedge t_j \wedge \ldots \wedge t_k$
  - $t_i = s_m \vee s_p \vee \ldots \vee s_q$: s_l son variables o sus negaciones
- Dada una expresión *e* en forma normal conjuntiva, ¿hay alguna asignación de valores a sus variables que haga *e* verdadera?
- Ejemplos: $e_1 = (!x_1 \vee x_2) \wedge (x_1 \vee x_3)$, $e_2 = (x_1 \vee x_2) \wedge !x_1 \wedge !x_2$
- MT estándar: $O(2^n)$
- MT no determinista: $O(n)$

MT y complejidad

Una MT acepta un lenguaje L en tiempo T(n) si cualquier cadena *w* de L ($|w| \leq n$) es aceptada en O(T(n)) movimientos

- Si la MT es no determinista, para cualquier cadena *w* de L existe al menos una secuencia de movimientos de longitud O(T(|*w*|)) que lleva a la aceptación

- Un lenguaje L pertenece a la clase TD(T(n)) si hay una MT ***multicinta determinista*** que acepta L en tiempo T(n)
- Un lenguaje L pertenece a la clase TND(T(n)) si hay una MT ***multicinta no determinista*** que acepta L en tiempo T(n)
  - $TD(T(n)) \subseteq TND(T(n))$

Complejidades P y NP

$$P = \bigcup_{i \geq 1} TD(n^i)$$

- Lenguajes aceptados por una MT **<u>determinista</u>** en tiempo polinómico

$$NP = \bigcup_{i \geq 1} TND(n^i)$$

- Lenguajes aceptados por una MT **<u>no determinista</u>** en tiempo polinómico

P $\subseteq$ NP

- ¿P = NP?

Problemas intratables: problemas computables pero que requerirían, para entradas grandes, tal cantidad de recursos (tiempo y memoria) que su implementación no es viable

- Los problemas de la clase P son tratables y el resto intratables (tesis de Cook-Karp)

Complejidades P y NP

Un lenguaje L_1 es reducible en tiempo polinómico a otro lenguaje L_2 si existe una MT determinista tal que cualquier cadena $w_1 \in \Sigma_1^+$ puede ser transformada en tiempo polinómico en otra cadena $w_2 \in \Sigma_2^+$ de tal forma que $w_1 \in L_1$ si y sólo si $w_2 \in L_2$

- Si L_1 es reducible en tiempo polinómico a L_2 y $L_2 \in P$, entonces $L_1 \in P$
- De igual forma, si $L_2 \in NP$, entonces $L_1 \in NP$

Un lenguaje L es NP-completo si $L \in NP$ y todo $L' \in NP$ es reducible en tiempo polinómico a L