Teoría de la Complejidad Computacional

Tema 3: Medidas abstractas de complejidad computacional

David Orellana Martín

Grupo de Investigación en Computación Natural
Dpto. Ciencias de la Computación e Inteligencia Artificial
Universidad de Sevilla

dorellana@us.es

Máster Universitario en Matemáticas

Curso 2023-2024

Índice

* Medidas de complejidad.
* Axiomas de **Blum**.
* Generación de medidas dinámicas de complejidad.
  - Teorema de **recursividad relativa**.
* Clases abstractas de complejidad computacional.
  - Teorema del **hueco** de **Borodin**.
* Existencia de **problemas intratables**.
* Teorema de **aceleración Blum**.

Se recomienda consultar el capítulo 14 del libro *"M.D. Davis, R. Sigal, E.J. Weyuker.* **Computability, Complexity and Languages. Fundamentals of Theoretical Computer Science**. Academic Press, Inc., Second Edition, 1994, XIX + 609 pages".

Medidas dinámicas de complejidad

Recursos computacionales necesarios para ejecutar un procedimiento mecánico:

- Medidas de complejidad.
  - Estática.
  - Dinámica.

Enumeración efectiva de las MTDs: $\{M_e \mid e \in \mathbb{N}\}$.

- $\varphi_e^{(n)}$: función de aridad $n \geqslant 1$ calculada por M_e.

$\{\varphi_e^{(n)} \mid e \in \mathbb{N}\}$: conjunto de las funciones **computables** n–arias por MTDs.

Resultados relevantes de Teoría de la Computabilidad

(a) Un **predicado computable** es una función total booleana que es computable.

(b) Un **conjunto** es **computable sii** su función característica es computable.

(c) Un **conjunto** es **recursivamente enumerable** (r.e.) **sii** es el dominio de una función computable.

(d) Todo conjunto computable es, asimismo, un conjunto r.e.

(e) Existen conjuntos r.e que **no** son computables; por ejemplo, el conjunto $\mathcal{K} = \{x \in \mathbb{N} \mid \varphi_x^{(1)}(x) \downarrow\}$ (conjunto del **problema de la parada**).

(f) Una función parcial es computable **sii** su gráfica es un conjunto r.e.

Medidas dinámicas de complejidad

Definición: Una **medida dinámica de complejidad computacional**[1] es un conjunto de funciones 1–arias $\{f_e \mid e \in \mathbb{N}\}$ de $\mathbb{N}$ en $\mathbb{N}$, tal que:

(a) Para cada $e \in \mathbb{N}$ se verifica: $\mathrm{dom}(f_e) = \mathrm{dom}(\varphi_e^{(1)})$.

(b) El predicado $\theta(e,x,y) = \left\{ \begin{array}{ll} 1 & \text{si } f_e(x) = y \\ 0 & \text{e.c.o.c.} \end{array} \right\}$ es **computable**.

Notaremos, brevemente, $\theta(e,x,y) \equiv (f_e(x) = y)$.

Las condiciones (a) y (b) se denominan **axiomas de Blum**.

[1] M. Blum. A Machine-Independent Theory of the Complexity of Recursive Functions. **Journal of the ACM**, **14**, 2 (1967), pages 322–336.

Manuel Blum

* Nació en Caracas el 26 de abril de 1938.
* Estudió en el MIT (Máster en Computer Science and Engineering).
* Doctor en Matemáticas (director: Marvin Minsky).
* Entre sus alumnos de doctorado, destacó L.M. Adleman.
* Premio Turing en 1995.

Medidas dinámicas de complejidad

En primer lugar, veamos que las funciones 1-arias que integran una medida de complejidad, son, todas ellas, computables.

Proposición: *Si* $\{f_e \mid e \in \mathbb{N}\}$ *es una medida de complejidad entonces, para cada* $e \in \mathbb{N}$ *se tiene que* f_e *es una función computable.*

> Basta tener presente que la gráfica de f_e, $G(f_e) = \{(x, y) \in \mathbb{N} \times \mathbb{N} \mid f_e(x) = y\}$, es un conjunto computable (por tanto, r.e) ya que para cada $(x, y) \in \mathbb{N} \times \mathbb{N}$ se verifica lo siguiente: $(x, y) \in G(f_e) \Leftrightarrow \theta(e, x, y) = 1$.

Medidas dinámicas de complejidad

Teorema: *Sea* $\{f_e \mid e \in \mathbb{N}\}$ *un conjunto de funciones 1–arias sobre* $\mathbb{N}$. *Son equivalentes:*

(a) *El predicado* $\theta(e,x,y) \equiv (f_e(x){=}y)$ *es computable.*

(b) *El predicado* $\theta'(e,x,y) \equiv (f_e(x){<}y)$ *es computable.*

(c) *El predicado* $\theta''(e,x,y) \equiv (f_e(x){\leqslant}\, y)$ *es computable.*

(a) ⇒ (b): Basta tener presente que

$\theta'(e,x,y) \equiv f_e(x) < y \Leftrightarrow \exists z < y\ (f_e(x) = z) \Leftrightarrow \exists z < y\ \theta(e,x,z)$

(b) ⇒ (c): Basta tener presente que

$\theta''(e,x,y) \equiv f_e(x) \leq y \Leftrightarrow f_e(x) < y+1 \Leftrightarrow\ \theta'(e,x,y+1)$

(c) ⇒ (a): Basta tener presente que

$\theta(e,x,y) \equiv f_e(x) = y \Leftrightarrow (y = 0 \wedge f_e(x) \leq 0) \vee (y \neq 0 \wedge f_e(x) \leq y \wedge \neg(f_e(x) \leq y-1))$

$\theta(e,x,y) \equiv (y = 0 \wedge \theta''(e,x,0)) \vee (y \neq 0 \wedge \theta''(e,x,y) \wedge \neg\theta''(e,x,y-1))$

El TIEMPO como medida de complejidad (I)

La **medida de complejidad TIEMPO** es el conjunto de funciones 1-arias sobre $\mathbb{N}$, $\{t_e \mid e \in \mathbb{N}\}$, definido como sigue:

$$t_e(x) = \begin{cases} \text{número de pasos en la computación } M_e(x), & \text{si } M_e(x)\downarrow \\ \uparrow & \text{, si } M_e(x)\uparrow \end{cases}$$

Este conjunto de funciones verifica, obviamente, el primer axioma de Blum.

Veamos que el conjunto $\{t_e \mid e \in \mathbb{N}\}$ satisface el segundo axioma de Blum.

Para ello, basta probar que el predicado $\theta''(e, x, y) \equiv (t_e(x) \leq y)$ es computable.

El TIEMPO como medida de complejidad (II)

Presentamos un esquema algorítmico que establece la computabilidad del predicado $\theta''(e, x, y) \equiv t_e(x) \leq y$.

Entrada: $(e, x, y) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}$

```
si y = 0 entonces devolver 0
si no
   t ← 1
   mientras t ≤ y hacer
    si la configuración C_t de M_e(x) es de parada entonces devolver 1
    si no
      t ← t + 1
   devolver 0
```

El ESPACIO como medida de complejidad (I)

La **medida de complejidad ESPACIO** es el conjunto de funciones 1-arias sobre $\mathbb{N}$, $\{s_e \mid e \in \mathbb{N}\}$, definido como sigue:

$$s_e(x) = \begin{cases} \text{mayor valor de cualquier variable en la computación } M_e(x), & \text{si } M_e(x)\downarrow \\ \uparrow & \text{, si } M_e(x)\uparrow \end{cases}$$

Este conjunto de funciones verifica, obviamente, el primer axioma de Blum.

Veamos que el conjunto $\{s_e \mid e \in \mathbb{N}\}$ satisface el segundo axioma de Blum.

Para ello, basta probar que el predicado $\theta''(e, x, y) \equiv (s_e(x) \leq y)$ es computable.

Notación: Si $M_e(x)\downarrow$ y $t \in \mathbb{N}$, notaremos $s_{e,t}(x)$ el mayor valor de cualquier variable en la configuración C_t de $M_e(x)$.

El ESPACIO como medida de complejidad (II)

Sea $(e, x, y) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}$ tal que $M_e(x)\downarrow$. Entonces es finito el conjunto de configuraciones C_t de $M_e(x)$ tal que $s_{e,t}(x) \leq y$. Sea $n(e, x, y)$ el cardinal de ese conjunto.

Entrada: $(e, x, y) \in \mathbb{N} \times \mathbb{N} \times \mathbb{N}$

$t \leftarrow 0$

`mientras` $t \leq n(e, x, y)$ `hacer`

  `si la configuración` C_t `de` $M_e(x)$ `verifica` $s_{e,t}(x) > y$ `entonces`

    `devolver 0`

  `si no`

    `si` C_t `es de parada entonces devolver 1`

    `si no`

      $t \leftarrow t + 1$

`devolver 0`

Nota: Obsérvese que si en la computación $M_e(x)$ existen dos configuraciones $C_t, C_{t'}$ (con $t \neq t'$) tales que $C_t = C_{t'}$, entonces $M_e(x)\uparrow$

Una clase de funciones que NO es una medida de complejidad

El conjunto de **todas** las **funciones computables** 1-arias $\{\varphi_e^{(1)} \mid e \in \mathbb{N}\}$ **no es** una medida de complejidad.

Basta probar que **no** se verifica el segundo axioma de Blum. En caso contrario, el predicado $\theta(e, x, y) \equiv (\varphi_e^{(1)}(x) = y)$ sería computable.

En tal situación, la función g de $\mathbb{N}$ en $\mathbb{N}$ definida por

$$g(x) = \begin{cases} 0 & \text{si } \varphi_x^{(1)}(x) \downarrow \\ \uparrow & \text{si } \varphi_x^{(1)}(x) \uparrow \end{cases}$$

sería computable pues su gráfica $\{(x, 0) \mid \varphi_x^{(1)}(x) \downarrow\} = \{(x, 0) \mid x \in \mathcal{K}\}$ sería un conjunto r.e.

Sea $e' \in \mathbb{N}$ tal que $g = \varphi_{e'}$. Entonces se verificaría:

$$\theta(e', x, 0) \equiv (\varphi_{e'}^{(1)}(x) = 0) \Leftrightarrow g(x) = 0 \Leftrightarrow x \in \mathcal{K}$$

Lo que contradice el supuesto de que el predicado θ fuese computable.

Generación de medidas de complejidad

Sea f una función parcial de $\mathbb{N}$ en $\mathbb{N}$. Si $f(x) \uparrow$ entonces notaremos $f(x) = +\infty$.

Definición: *Un* **factor de escala** *es una función computable total 1–aria, h, tal que*

- *h es creciente: para cada* $x \in \mathbb{N}$ *se tiene* $h(x) \leqslant h(x+1)$.
- $\lim_{x \to +\infty} h(x) = +\infty$.

Teorema: *Si* $\{f_e \mid e \in \mathbb{N}\}$ *es una medida de complejidad y h es un factor de escala, entonces* $\{h \circ f_e \mid e \in \mathbb{N}\}$ *es otra medida de complejidad.*

- Una demostración de este resultado se puede consultar en el texto recomendado (**teorema 1.1, páginas 421-422**).

Los factores de escala proporcionan un **mecanismo** para **generar** nuevas medidas de complejidad, a partir de otras medidas.

Definición: *Un* **predicado 1–ario**, $\theta(x)$, *sobre* $\mathbb{N}$ **se verifica asintóticamente** *sii existe* $n \in \mathbb{N}$ *tal que* $\forall x \geqslant n \ (\theta(x) = 1)$.

Teorema: (**Recursividad relativa**) *Sean* $\{f_e \mid \ e \in \mathbb{N}\}$ *y* $\{g_e \mid \ e \in \mathbb{N}\}$ *medidas de complejidad. Existe una función,* $h : \mathbb{N}^2 \to \mathbb{N}$, *computable total verificando:*

(a) $\forall x \ \forall y \ (h(x, y) < h(x, y + 1))$.

(b) $\forall e \ \exists n \ \forall x \geqslant n \ (f_e(x) \leqslant h(x, g_e(x)))$.

(c) $\forall e \ \exists n \ \forall x \geqslant n \ (g_e(x) \leqslant h(x, f_e(x)))$.

(*Notación:* $h(x, +\infty) = +\infty$, *para cada* $x \in \mathbb{N}$)

- Una demostración de este resultado se puede consultar en el texto recomendado (**teorema 1.2, páginas 422-423**).

Dos medidas de complejidad arbitrarias siempre están relacionadas entre sí; específicamente, mediante una "especie de factor de escala" (asintóticamente).

Clases abstractas de complejidad

Definición: *Sea* $\mathcal{M} = \{f_e \mid e \in \mathbb{N}\}$ *una medida de complejidad y* g *una función computable total 1–aria. La* **clase de complejidad** *determinada por* g*, respecto de* $\mathcal{M}$ *es:*

$$\mathcal{C}_g^{\mathcal{M}} = \{\varphi_e \mid \varphi_e \text{ total} \wedge \exists n_0 \forall x \geqslant n_0 \ (f_e(x) \leqslant g(x))\}$$

Cuestión: Si $g < h$ (asintóticamente) ¿se puede asegurar que $\mathcal{C}_g^{\mathcal{M}} \subsetneq \mathcal{C}_h^{\mathcal{M}}$?

Teorema (del **hueco, Borodin**): *Sea* $\{f_e \mid e \in \mathbb{N}\}$ *una medida de complejidad. Sea* $g(x, y)$ *computable total tal que* $\forall x, y \ (g(x, y) > y)$*. Existe una función total computable* h *de* $\mathbb{N}$ *en* $\mathbb{N}$ *verificando:*

$$\forall e \ \forall x > e \ (f_e(x)\downarrow \wedge f_e(x) < g(x, h(x)) \Longrightarrow f_e(x) \leqslant h(x))$$

* Una demostración de este resultado se puede consultar en el texto recomendado (**teorema 2.1, páginas 426-427**).

Corolario: *Sea $\mathcal{M}$ una medida de complejidad y $r(x)$ una función computable total tal que $\forall x\ (r(x) \geqslant x)$. Existe una función computable total, $h : \mathbb{N} \to \mathbb{N}$, verificando que $\mathcal{C}^{\mathcal{M}}_{r \circ h} = \mathcal{C}^{\mathcal{M}}_{h}$.*

Consecuencia interesante:

- **Existe una función computable** *total, $h'(x)$, verificando lo siguiente*:
  - Toda MTD cuyo tiempo de ejecución sobre una máquina "rápida" está acotado por $h'(x)$, asintóticamente, también tardará un tiempo acotado por $h'(x)$, asintóticamente, cuando se ejecuta sobre **una máquina muchísmo más lenta** que la anterior.

Existencia de problemas intratables

Teorema: *Sea* $\{f_e \mid \ e \in \mathbb{N}\}$ *una medida de complejidad. Para cada función computable total 1–aria,* f*, existe una función computable 1-aria,* g*, verificando que si* $e \in \mathbb{N}$ *satisface que* $g = \varphi_e^{(1)}$*, entonces*

$$\exists n \, \forall x \geqslant n \, (f(x) < f_e(x))$$

* Una demostración de este resultado se puede consultar en la sección **Material de trabajo** de la página web de la asignatura.

Interpretación computacional: Respecto de cualquier medida de complejidad, existe un problema X que es **intratable** respecto de dicha medida

(cualquier solución mecánica de X consume una cantidad "arbitrariamente grande" de recursos, para ejemplares de "tamaño *grande*").

Existencia de problemas que carecen de soluciones óptimas

Teorema: (de **aceleración** de **Blum**) *Sea* $\{f_e \mid e \in \mathbb{N}\}$ *una medida de complejidad y* g *una función computable total 1–aria y creciente. Existe una función computable total 1–aria,* h*, verificando:*

- **Para cada MTD,** M_e**, que calcula** h**, existe otra MTD,** $M_{e'}$**, que también calcula** h **y, además, verifica:** $\exists n\ \forall x \geqslant n\ (g(f_{e'}(x)) < f_e(x))$**.**

* Una demostración de este resultado se puede consultar en el texto recomendado (**teorema 4.3, páginas 435-438**).

Corolario: *Sea* $\{f_e \mid e \in \mathbb{N}\}$ *una medida de complejidad. Existe una función computable total 1–aria que* **carece** *de una* **MTD óptima** *que la calcule.*

Interpretación computacional del teorema de aceleración

Respecto de cualquier medida abstracta de complejidad, existen problemas que carecen de una *MTD óptima* que lo resuelve.

Consecuencia interesante: Sean A (**rapidísima**) y B (**lentísima**) dos máquinas que implementan MTDs. Existe un problema tal que para cada MTD, M_e, que lo resuelva existe otra MTD, $M_{e'}$, que también lo resuelve y, además,

* **$M_{e'}$ ejecutada sobre la máquina lenta B es estrictamente más rápida que M_e ejecutada sobre la máquina rápida A (asintóticamente).**

Sean A y B dos máquinas tales que la primera es 10^6 más rápida que la segunda

- Si para cada MTD M_e, denotamos $t_e^A(x)$ y $t_e^B(x)$ los tiempos necesarios para ejecutar $M_e(x)$ sobre las máquinas A y B, respectivamente, entonces $t_e^A(x) = \frac{t_e^B(x)}{10^6}$.
- Se aplica el teorema de aceleración de Blum a la medida de complejidad $\{t_e^B \mid e \in \mathbb{N}\}$ y a la función computable total $g(x) = 10^6 x$.
- Entonces existe una función computable total h de $\mathbb{N}$ en $\mathbb{N}$ tal que
  - Para cada MTD M_e que calcula h existe otra MTD $M_{e'}$ que también calcula h y, además, verifica que existe $n_1 \in \mathbb{N}$ verificando $g(t_{e'}^B(x)) < t_e^B(x)$, para cada $x \geqslant n_1$. Luego

$$t_{e'}^B(x) < t_e^A(x), \text{ para cada } x \geqslant n_1$$

Existe un problema (representado por h) que para cada solución mecánica M_e del mismo, existe otra solución mecánica $M_{e'}$ que, ejecutada sobre la máquina lentísima B, tarda menos que la solución M_e ejecutada sobre la máquina rapidísima A.