Introducción a la Computación (Matemática)

Primer Cuatrimestre de 2018

Complejidad Algorítmica

Ejemplo 1.1

Encabezado: $E_1 : A \in \mathbb{Z}[\,] \to \mathbb{Z}$

Precondición: $\{A = A_0\}$

Poscondición: $\{RV = (\#i \in \mathbb{Z})\ 0 \leq i < |A_0| \ \wedge \ A_0[i] = \sum_{0 \leq j < i} A_0[j]\}$

Ejemplo 1.1

Encabezado: $E_1 : A \in \mathbb{Z}[\,] \to \mathbb{Z}$

Precondición: $\{A = A_0\}$

Poscondición: $\{RV = (\#i \in \mathbb{Z})\ 0 \leq i < |A_0| \ \wedge\ A_0[i] = \sum_{0 \leq j < i} A_0[j]\}$

```
RV ← 0
i ← 0
while (i < |A|) {
        j ← 0
        sumaAnteriores ← 0
        while (j < i) {
                sumaAnteriores ← sumaAnteriores + A[j]
                j ← j + 1
        }
        if (sumaAnteriores = A[i]) {
           RV ← RV + 1
        }
        i ← i + 1
}
```

Ejemplo 1.2

Encabezado: $E_1 : A \in \mathbb{Z}[\,] \to \mathbb{Z}$

Precondición: $\{A = A_0\}$

Poscondición: $\{RV = (\#i \in \mathbb{Z})\ 0 \leq i < |A_0| \ \wedge \ A_0[i] = \sum_{0 \leq j < i} A_0[j]\}$

$RV \leftarrow 0$

$i \leftarrow 0$

$sumaAnteriores \leftarrow 0$

while $(i < |A|)$ $\{$

  if $(sumaAnteriores = A[i])$ $\{$

    $RV \leftarrow RV + 1$

  $\}$

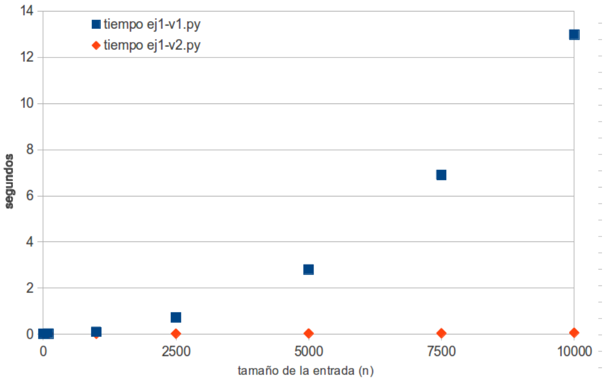
  $sumaAnteriores \leftarrow sumaAnteriores + A[i]$

  $i \leftarrow i + 1$

$\}$

Tiempo de ejecución

n	tiempo ej1-v1.py	tiempo ej1-v2.py
1	0.027	0.028
10	0.029	0.029
100	0.034	0.024
1000	0.107	0.027
2500	0.736	0.031
5000	2.808	0.039
7500	6.901	0.048
10000	12.98	0.07



Comparación de los tiempos de ejecución de implementaciones en Python de los algoritmos de los ejemplos 1.1 (*ej1-v1.py*) y 1.2 (*ej1-v2.py*).

Tiempo de ejecución (otro enfoque)

El tiempo de ejecución de un programa se mide en función del tamaño de la entrada.

- Ejemplo: longitud de la lista de entrada.

Tiempo de ejecución (otro enfoque)

El tiempo de ejecución de un programa se mide en función del tamaño de la entrada.

- Ejemplo: longitud de la lista de entrada.

Notación: $T(n)$: tiempo de ejecución de un programa con una entrada de tamaño n.

- Unidad: cantidad de instrucciones.
- Ejemplo: $T(n) = c \cdot n^2$, donde c es una constante.

Tiempo de ejecución

Podemos considerar tres casos del tiempo de ejecución:

- peor caso: tiempo máximo de ejecución para alguna entrada;
- mejor caso: tiempo mínimo de ejecución para alguna entrada;
- caso promedio: tiempo de ejecución para la *entrada promedio*.

Tiempo de ejecución

Podemos considerar tres casos del tiempo de ejecución:

- peor caso: tiempo máximo de ejecución para alguna entrada;
- mejor caso: tiempo mínimo de ejecución para alguna entrada;
- caso promedio: tiempo de ejecución para la *entrada promedio*.

Vamos a ver sólo el peor caso: $T(n)$ es una cota superior del tiempo de ejecución para entradas arbitrarias de tamaño n.

Cálculo del tiempo de ejecución

Instrucciones minimales: acceso a una variable (asignación o consulta) y operaciones simples de tipos básicos. $T_S(n) = 1$

Cálculo del tiempo de ejecución

Instrucciones minimales: acceso a una variable (asignación o consulta) y operaciones simples de tipos básicos. $T_S(n) = 1$

- Ej: $T_{x \leftarrow 2*y+1}(n) = T_y + T_* + T_+ + T_\leftarrow = 1+1+1+1 = 4$

Cálculo del tiempo de ejecución

Instrucciones minimales: acceso a una variable (asignación o consulta) y operaciones simples de tipos básicos. $T_S(n) = 1$

- Ej: $T_{x \leftarrow 2*y+1}(n) = T_y + T_* + T_+ + T_\leftarrow = 1+1+1+1 = 4$

Secuencialización: $T_{S_1;S_2}(n) = T_{S_1}(n) + T_{S_2}(n)$

Cálculo del tiempo de ejecución

Instrucciones minimales: acceso a una variable (asignación o consulta) y operaciones simples de tipos básicos. $T_S(n) = 1$

- Ej: $T_{x \leftarrow 2*y+1}(n) = T_y + T_* + T_+ + T_\leftarrow = 1+1+1+1 = 4$

Secuencialización: $T_{S_1;S_2}(n) = T_{S_1}(n) + T_{S_2}(n)$

- Ej: $T_{x \leftarrow y+1;\, y \leftarrow 0}(n) = T_{x \leftarrow y+1}(n) + T_{y \leftarrow 0}(n) = 3 + 1 = 4$

Cálculo del tiempo de ejecución

Instrucciones minimales: acceso a una variable (asignación o consulta) y operaciones simples de tipos básicos. $T_S(n) = 1$

- Ej: $T_{x \leftarrow 2*y+1}(n) = T_y + T_* + T_+ + T_\leftarrow = 1{+}1{+}1{+}1 = 4$

Secuencialización: $T_{S_1;S_2}(n) = T_{S_1}(n) + T_{S_2}(n)$

- Ej: $T_{x \leftarrow y+1;\, y \leftarrow 0}(n) = T_{x \leftarrow y+1}(n) + T_{y \leftarrow 0}(n) = 3 + 1 = 4$

Condicional:

$T_{\text{if } (B)\ S_1 \text{ else } S_2}(n) = T_B(n) + \text{máx}(T_{S_1}(n), T_{S_2}(n))$

Cálculo del tiempo de ejecución

Instrucciones minimales: acceso a una variable (asignación o consulta) y operaciones simples de tipos básicos. $T_S(n) = 1$

- Ej: $T_{x \leftarrow 2*y+1}(n) = T_y + T_* + T_+ + T_\leftarrow = 1{+}1{+}1{+}1 = 4$

Secuencialización: $T_{S_1;S_2}(n) = T_{S_1}(n) + T_{S_2}(n)$

- Ej: $T_{x \leftarrow y+1;\, y \leftarrow 0}(n) = T_{x \leftarrow y+1}(n) + T_{y \leftarrow 0}(n) = 3 + 1 = 4$

Condicional:

$T_{\text{if } (B)\ S_1 \text{ else } S_2}(n) = T_B(n) + \max(T_{S_1}(n), T_{S_2}(n))$

Ciclo: $T_{\text{while } (B)\ S}(n) = T_B(n) + \Sigma_{i \in \text{Iteraciones}} \left(T_{S_i}(n) + T_B(n)\right)$

Ejemplo 1.1

$RV \leftarrow 0$
$i \leftarrow 0$
while $(i < |A|)$ {
    $j \leftarrow 0$
    $sumaAnteriores \leftarrow 0$
    while $(j < i)$ {
        $sumaAnteriores \leftarrow sumaAnteriores + A[j]$
        $j \leftarrow j + 1$
    }
    if $(sumaAnteriores = A[i])$ {
        $RV \leftarrow RV + 1$
    }
    $i \leftarrow i + 1$
}

Ejemplo 1.1

$RV \leftarrow 0$ (1)
$i \leftarrow 0$ (1)
while $(i < |A|)$ { (3)
    $j \leftarrow 0$ (1)
    $sumaAnteriores \leftarrow 0$ (1)
    while $(j < i)$ { (3)
        $sumaAnteriores \leftarrow sumaAnteriores + A[j]$ (5)
        $j \leftarrow j + 1$ (3)
    }
    if $(sumaAnteriores = A[i])$ { (4)
        $RV \leftarrow RV + 1$ (3)
    }
    $i \leftarrow i + 1$ (3)
}

Ejemplo 1.1

$RV \leftarrow 0$ (1)
$i \leftarrow 0$ (1)
while $(i < |A|)$ { (3)
  $j \leftarrow 0$ (1)
  $sumaAnteriores \leftarrow 0$ (1)
  while $(j < i)$ { (3)
    $sumaAnteriores \leftarrow sumaAnteriores + A[j]$ (5)
    $j \leftarrow j + 1$ (3)
  }
  if $(sumaAnteriores = A[i])$ { (4)
    $RV \leftarrow RV + 1$ (3)
  }
  $i \leftarrow i + 1$ (3)
}

$$T(|A|) = 1+1+3+\Sigma_{0\leq i<|A|}\big(1+1+3+\Sigma_{0\leq j<i}(5+3+3)+4+3+3+3\big)$$

Ejemplo 1.1

$RV \leftarrow 0$ (1)
$i \leftarrow 0$ (1)
while $(i < |A|)$ { (3)
  $j \leftarrow 0$ (1)
  $sumaAnteriores \leftarrow 0$ (1)
  while $(j < i)$ { (3)
    $sumaAnteriores \leftarrow sumaAnteriores + A[j]$ (5)
    $j \leftarrow j + 1$ (3)
  }
  if $(sumaAnteriores = A[i])$ { (4)
    $RV \leftarrow RV + 1$ (3)
  }
  $i \leftarrow i + 1$ (3)
}

$$
\begin{aligned}
T(|A|) &= 1+1+3+\Sigma_{0\leq i<|A|}\left(1+1+3+\Sigma_{0\leq j<i}(5+3+3)+4+3+3+3\right)\\
&= 5+\Sigma_{0\leq i<|A|}\left(18+\Sigma_{0\leq j<i}11\right) = 5+18\,|A|+\Sigma_{0\leq i<|A|}i\cdot 11
\end{aligned}
$$

Ejemplo 1.1

$RV \leftarrow 0$ (1)
$i \leftarrow 0$ (1)
while $(i < |A|)$ { (3)
  $j \leftarrow 0$ (1)
  $sumaAnteriores \leftarrow 0$ (1)
  while $(j < i)$ { (3)
    $sumaAnteriores \leftarrow sumaAnteriores + A[j]$ (5)
    $j \leftarrow j + 1$ (3)
  }
  if $(sumaAnteriores = A[i])$ { (4)
    $RV \leftarrow RV + 1$ (3)
  }
  $i \leftarrow i + 1$ (3)
}

$$\begin{aligned} T(|A|) &= 1+1+3+\Sigma_{0\leq i<|A|}\left(1+1+3+\Sigma_{0\leq j<i}(5+3+3)+4+3+3+3\right) \\ &= 5+\Sigma_{0\leq i<|A|}\left(18+\Sigma_{0\leq j<i}11\right) = 5+18\,|A|+\Sigma_{0\leq i<|A|}i\cdot 11 \\ &= 5+18\,|A|+\frac{11}{2}(|A|-1)|A| = 5+\frac{25}{2}\,|A|+\frac{11}{2}\,|A|^2 \end{aligned}$$

Ejemplo 1.2

$RV \leftarrow 0$
$i \leftarrow 0$
$sumaAnteriores \leftarrow 0$
while $(i < |A|)$ $\{$
    if $(sumaAnteriores = A[i])$ $\{$
        $RV \leftarrow RV + 1$
    $\}$
    $sumaAnteriores \leftarrow sumaAnteriores + A[i]$
    $i \leftarrow i + 1$
$\}$

Ejemplo 1.2

$RV \leftarrow 0$ (1)
$i \leftarrow 0$ (1)
$sumaAnteriores \leftarrow 0$ (1)
while $(i < |A|)$ { (3)
  if $(sumaAnteriores = A[i])$ { (4)
    $RV \leftarrow RV + 1$ (3)
  }
  $sumaAnteriores \leftarrow sumaAnteriores + A[i]$ (5)
  $i \leftarrow i + 1$ (3)
}

Ejemplo 1.2

$RV \leftarrow 0$ (1)
$i \leftarrow 0$ (1)
$sumaAnteriores \leftarrow 0$ (1)
while $(i < |A|)$ { (3)
    if $(sumaAnteriores = A[i])$ { (4)
        $RV \leftarrow RV + 1$ (3)
    }
    $sumaAnteriores \leftarrow sumaAnteriores + A[i]$ (5)
    $i \leftarrow i + 1$ (3)
}

$$\begin{aligned} T(|A|) &= 1 + 1 + 1 + 3 + \Sigma_{0 \leq i < |A|} \left(4 + 3 + 5 + 3 + 3\right) \\ &= 6 + \Sigma_{0 \leq i < |A|} 18 = 6 + 18\,|A| \end{aligned}$$

Orden del tiempo de ejecución

Definición: Decimos que $T(n) \in O(f(n))$ si existen constantes enteras positivas c y n_0 tales que para $n \geq n_0$, $T(n) \leq c \cdot f(n)$.

Ejemplo: $T(n) = 3n^3 + 2n^2$.

Orden del tiempo de ejecución

Definición: Decimos que $T(n) \in O(f(n))$ si existen constantes enteras positivas c y n_0 tales que para $n \geq n_0$, $T(n) \leq c \cdot f(n)$.

Ejemplo: $T(n) = 3n^3 + 2n^2$.

$T(n) \in O(n^3)$, dado que si tomamos $n_0 = 1$ y $c = 5$, vale que para $n \geq 1$, $T(n) \leq 5 \cdot n^3$.

Orden del tiempo de ejecución

Definición: Decimos que $T(n) \in O(f(n))$ si existen constantes enteras positivas c y n_0 tales que para $n \geq n_0$, $T(n) \leq c \cdot f(n)$.

Ejemplo: $T(n) = 3n^3 + 2n^2$.

$T(n) \in O(n^3)$, dado que si tomamos $n_0 = 1$ y $c = 5$, vale que para $n \geq 1$, $T(n) \leq 5 \cdot n^3$.

Ejemplo 1.1: $T(|A|) = 5 + \frac{25}{2}|A| + \frac{11}{2}|A|^2 \ \in O(|A|^2)$ (orden cuadrático)

Orden del tiempo de ejecución

Definición: Decimos que $T(n) \in O(f(n))$ si existen constantes enteras positivas c y n_0 tales que para $n \geq n_0$, $T(n) \leq c \cdot f(n)$.

Ejemplo: $T(n) = 3n^3 + 2n^2$.

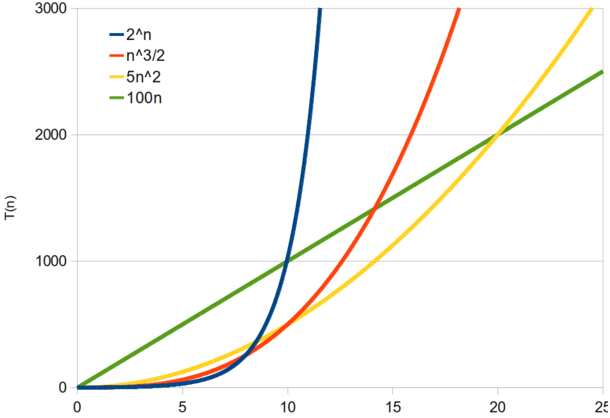
$T(n) \in O(n^3)$, dado que si tomamos $n_0 = 1$ y $c = 5$, vale que para $n \geq 1$, $T(n) \leq 5 \cdot n^3$.

Ejemplo 1.1: $T(|A|) = 5 + \frac{25}{2}\,|A| + \frac{11}{2}\,|A|^2 \; \in O(|A|^2)$ (orden cuadrático) **Ejemplo 1.2:**

$T(|A|) = 6 + 18\,|A| \; \in O(|A|)$ (orden lineal)

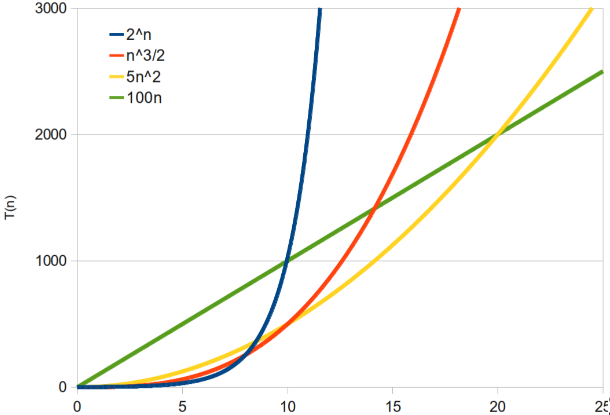
Entonces, ¿cuál programa usamos?

Atención #1: Tamaño de la entrada



Entonces, ¿cuál programa usamos?

Atención #1: Tamaño de la entrada



Si el tamaño de la entrada está acotado, quizá conviene usar un programa de mayor orden pero con constantes bajas.

Entonces, ¿cuál programa usamos?

Atención #2: Objetivos contrapuestos

Para resolver un problema, queremos un programa...

1. que sea fácil de programar (que escribirlo nos demande poco tiempo, que sea simple y fácil de entender);
2. que consuma pocos recursos: tiempo y espacio (memoria, disco rígido).

Entonces, ¿cuál programa usamos?

Atención #2: Objetivos contrapuestos

Para resolver un problema, queremos un programa...

1. que sea fácil de programar (que escribirlo nos demande poco tiempo, que sea simple y fácil de entender);
2. que consuma pocos recursos: tiempo y espacio (memoria, disco rígido).

En general priorizamos un objetivo sobre el otro:

- para programas que correrán pocas veces, priorizamos el objetivo 1;
- para programas que correrán muchas veces, priorizamos el objetivo 2.

Repaso de la clase de hoy

- Tiempo de ejecución: en segundos y en cantidad de operaciones.
- Peor caso, mejor caso y caso promedio.
- Cálculo del tiempo de ejecución.
- Orden del tiempo de ejecución.

Próximos temas

- Algoritmos de búsqueda.
- Algoritmos de ordenamiento.