

Programación Dinámica Bottom-Up

Julián Braier

FCEN - UBA

1c 2024

Problema del CD otra vez

- ▶ **Problema:** dada K la capacidad del CD (en segundos) y una lista $d = (d_1, \dots, d_n)$ con las duraciones de n canciones (en segundos). Cuál es la máxima suma de las duraciones de las canciones que agregamos que podemos lograr (obviamente que sin exceder la capacidad del CD). No vale ni cortar ni repetir canciones.

Problema del CD otra vez

- ▶ **Problema:** dada K la capacidad del CD (en segundos) y una lista $d = (d_1, \dots, d_n)$ con las duraciones de n canciones (en segundos). Cuál es la máxima suma de las duraciones de las canciones que agregamos que podemos lograr (obviamente que sin exceder la capacidad del CD). No vale ni cortar ni repetir canciones.
- ▶ **Ejemplo:** si $k = 8$ y $d = [2, 4, 5]$ la respuesta es

Problema del CD otra vez

- ▶ **Problema:** dada K la capacidad del CD (en segundos) y una lista $d = (d_1, \dots, d_n)$ con las duraciones de n canciones (en segundos). Cuál es la máxima suma de las duraciones de las canciones que agregamos que podemos lograr (obviamente que sin exceder la capacidad del CD). No vale ni cortar ni repetir canciones.
- ▶ **Ejemplo:** si $k = 8$ y $d = [2, 4, 5]$ la respuesta es 7.

Problema del CD otra vez

- ▶ Demos una formulación recursiva para resolver el problema.

Problema del CD otra vez

- ▶ Demos una formulación recursiva para resolver el problema.
- ▶ $CD(i, k)$: "máxima duración que puedo lograr eligiendo solamente canciones del prefijo (d_1, \dots, d_i) en un CD de capacidad k ".

Problema del CD otra vez

- ▶ Demos una formulación recursiva para resolver el problema.
- ▶ $CD(i, k)$: "máxima duración que puedo lograr eligiendo solamente canciones del prefijo (d_1, \dots, d_i) en un CD de capacidad k ".

$CD(i, k) =$

$$\begin{cases} 0 & \text{si } i = 0 \\ CD(i - 1, k) & \text{si } i \neq 0 \wedge d_i \leq k \\ \max(CD(i - 1, k), CD(i - 1, k - d_i) + d_i) & \text{cc} \end{cases}$$

Solución recursiva en C++

```
int CD(int i, int k){  
    if(i == 0) return 0;  
    if(duracion[i] > k) return CD(i - 1, k);  
    else return max(CD(i - 1, k), CD(i - 1, k - duracion[i]) + duracion[i]);  
}
```

Solución Top-Down

- ▶ Usamos matriz $M[0..n][0..K]$ inicializada en \perp .

Solución Top-Down

- ▶ Usamos matriz $M[0..n][0..K]$ inicializada en \perp .
- ▶ Hay que chequear si ya tenemos guardado en el diccionario el valor que nos piden. Si no lo tenemos lo calculamos y lo guardamos. Después retornamos el valor del diccionario.

Solución Top-Down

```
int CD(int i, int k){  
    if(i == 0) return 0;  
    if(duracion[i] > k) return CD(i - 1, k);  
    else return max(CD(i - 1, k), CD(i - 1, k - duracion[i]) + duracion[i]);  
}
```

Figure: Solución recursiva

```
int CD(int i, int k){  
    if(i == 0) return 0;  
    if(M[i][k] == BOTTOM){  
        if(duracion[i] > k) M[i][k] = CD(i - 1, k);  
        else M[i][k] = max(CD(i - 1, k), CD(i - 1, k - duracion[i]) + duracion[i]);  
    }  
    return M[i][k];  
}
```

Figure: Solución Top-Down

Bottom-Up

- ▶ En lugar de usar recursión usamos iteración.

Bottom-Up

- ▶ En lugar de usar recursión usamos iteración.
- ▶ Iterativamente resolvemos todos los subproblemas, guardando los resultados en la estructura de memoización.

Bottom-Up

- ▶ En lugar de usar recursión usamos iteración.
- ▶ Iterativamente resolvemos todos los subproblemas, guardando los resultados en la estructura de memoización.
- ▶ En qué orden?

Solución Bottom-Up

```
int CD(int i, int k){  
    if(i == 0) return 0;  
    if(M[i][k] == BOTTOM){  
        if(duracion[i] > k) M[i][k] = CD(i - 1, k);  
        else M[i][k] = max(CD(i - 1, k), CD(i - 1, k - duracion[i]) + duracion[i]);  
    }  
    return M[i][k];  
}
```

Figure: Solución Top-Down

```
int CD(int n, int K){  
    for(int i = 0; i <= n; i++){  
        for(int k = 0; k <= K; k++){  
            if(i == 0) M[i][k] = 0;  
            else if(duracion[i] > k) M[i][k] = M[i-1][k];  
            else M[i][k] = max(M[i-1][k], M[i-1][k - duracion[i]] + duracion[i]);  
        }  
    }  
    return M[n][K];  
}
```

Figure: Solución Bottom-Up

Ahormando memoria

N y M son dos arreglos de enteros, de tamaño $K + 1$.

```
int CD(int n, int K){  
    for(int k = 0; k <= K; k++)  
        M[k] = 0;  
    for(int i = 1; i <= n; i++){  
        for(int k = 0; k <= K; k++){  
            if(duracion[i] > k) N[k] = M[k];  
            else N[k] = max(M[k], M[k - duracion[i]] + duracion[i]);  
        }  
        swap(&M, &N);  
    }  
    return M[K];  
}
```

Figure: Solución Bottom-Up con $O(K)$ memoria.

Ahormando memoria bis

```
int CD(int n, int K){  
    for(int k = 0; k <= K; k++)  
        M[k] = 0;  
    for(int i = 1; i <= n; i++)  
        for(int k = K; k >= 0; k--)  
            if(duracion[i] <= k) M[k] = max(M[k], M[k - duracion[i]] + duracion[i]);  
    return M[K];  
}
```

Figure: Solución Bottom-Up, también con $O(K)$ memoria, más canchero.

Ahormando memoria bis

```
int CD(int n, int K){  
    for(int k = 0; k <= K; k++)  
        M[k] = 0;  
    for(int i = 1; i <= n; i++)  
        for(int k = K; k >= 0; k--)  
            if(duracion[i] <= k) M[k] = max(M[k], M[k - duracion[i]] + duracion[i]);  
    return M[K];  
}
```

Figure: Solución Bottom-Up, también con $O(K)$ memoria, más canchero.

Y si ahora quiero reconstruir la solución?

Mi cuatrimestre como estudiante en Algo III

2c2019	4.4	4.3	3.9	3.6	4.1	4.8	4.0	4.6	4.3	4.2	4.4	4.3	4.1	4.2	4.3	4.4	4.2	4.6	46 alumnos - ver docentes más detalles ver 15 comentarios
1c2019	4.4	4.4	3.6	3.5	3.6	4.3	4.4	4.3	4.2	4.3	3.9	3.9	3.7	3.8	3.9	4.7	4.0	4.6	94 alumnos - ver docentes más detalles ver 35 comentarios
2c2018	4.4	3.9	4.1	4.0	3.9	4.0	4.4	4.4	4.5	4.3	4.6	4.4	4.2	4.3	4.4	4.9	4.3	4.4	44 alumnos - ver docentes más detalles ver 10 comentarios
1c2018	4.6	4.7	4.1	3.9	4.0	4.8	4.8	4.0	4.4	4.3	4.5	4.4	4.0	4.1	4.0	4.7	4.3	59 alumnos - ver docentes más detalles ver 18 comentarios	
2c2017	4.6	4.7	4.2	4.1	4.2	4.8	4.8	4.6	4.6	4.7	4.4	4.0	4.2	4.2	4.1	4.9	4.4	4.4	55 alumnos - ver docentes más detalles ver 19 comentarios
1c2017	4.6	4.6	4.2	4.0	4.3	4.8	4.8	4.2	4.3	4.2	4.5	4.2	4.1	4.2	4.2	4.7	4.4	4.4	75 alumnos - ver docentes más detalles ver 17 comentarios
2c2016	4.8	4.8	4.4	4.2	4.5	4.8	4.8	4.5	4.8	4.8	4.7	4.7	4.6	4.7	4.7	4.8	4.6	57 alumnos - ver docentes más detalles ver 17 comentarios	
1c2016	4.6	4.7	4.3	3.9	4.4	4.9	4.8	4.6	4.7	4.7	4.5	4.7	4.3	4.3	4.7	4.8	4.5	64 alumnos - ver docentes más detalles ver 19 comentarios	
2c2015	4.7	4.7	4.1	3.7	4.4	4.9	4.8	4.4	4.5	4.7	4.4	4.1	4.1	4.5	3.9	4.7	4.4	4.4	66 alumnos - ver docentes más detalles ver 19 comentarios
1c2015	4.7	4.6	3.9	3.7	4.0	4.6	4.7	4.2	4.4	4.4	4.4	4.2	4.2	4.3	4.2	4.6	4.3	74 alumnos - ver docentes más detalles ver 26 comentarios	
2c2014	4.4	4.3	3.7	3.4	4.2	4.5	4.6	4.2	4.4	4.4	4.2	4.1	3.9	4.0	3.7	4.6	4.1	70 alumnos - ver docentes más detalles ver 21 comentarios	
1c2014	4.5	4.6	3.8	3.3	3.9	4.7	4.6	4.3	4.2	4.3	4.3	4.3	3.9	4.1	3.9	4.6	4.2	51 alumnos - ver docentes más detalles ver 27 comentarios	
2c2013	4.6	4.5	3.7	3.4	4.2	4.7	4.8	4.1	4.4	4.4	4.2	4.0	3.7	3.8	3.7	4.7	4.1	55 alumnos - ver docentes más detalles ver 16 comentarios	
1c2013	4.2	4.4	3.7	3.4	4.0	4.6	4.5	4.2	4.3	4.4	4.2	4.2	3.8	3.9	3.8	4.6	4.1	73 alumnos - ver docentes más detalles ver 29 comentarios	
2c2012	4.3	4.3	3.7	3.5	4.1	4.5	4.5	4.3	4.2	4.3	4.4	3.9	4.0	4.2	4.2	4.6	4.1	35 alumnos - ver docentes más detalles ver 12 comentarios	
1c2012	4.3	4.6	4.1	3.8	4.1	4.6	4.6	4.1	4.7	4.7	4.4	4.0	4.1	4.1	4.0	4.7	4.3	62 alumnos - ver docentes más detalles ver 34 comentarios	
2c2011	4.6	4.6	3.9	3.5	3.7	4.0	4.8	3.8	4.5	4.5	4.5	4.2	4.3	4.4	4.2	4.6	4.3	32 alumnos - ver docentes más detalles ver 15 comentarios	
1c2011	4.1	4.3	3.6	3.4	3.7	4.0	4.2	4.2	4.0	4.0	4.0	3.9	3.7	3.9	4.0	4.6	4.0	36 alumnos - ver docentes más detalles ver 18 comentarios	
2c2010	4.4	4.5	3.6	3.8	3.6	4.6	4.7	4.2	4.3	4.5	4.2	4.1	3.8	4.0	3.9	4.6	4.1	47 alumnos - ver docentes más detalles ver 16 comentarios	

1



Teóricas

- Francisco Soulignac (Prof)

Práctica

- Alejandro Strejilevich de Loma (JTP)
- Brian Curcio (JTP)
- Carolina Lang (AY2)
- Sebastián Taboh (AY2)
- Alejandra Bringas (AY2)

Laboratorio

- Federico Pousa (JTP)
- Gonzalo Lera Romero (AY1)
- Florencia Zanollo (AY2)
- Jonathan Seijo (AY2)

2

2019 - 1er. Cuatrimestre

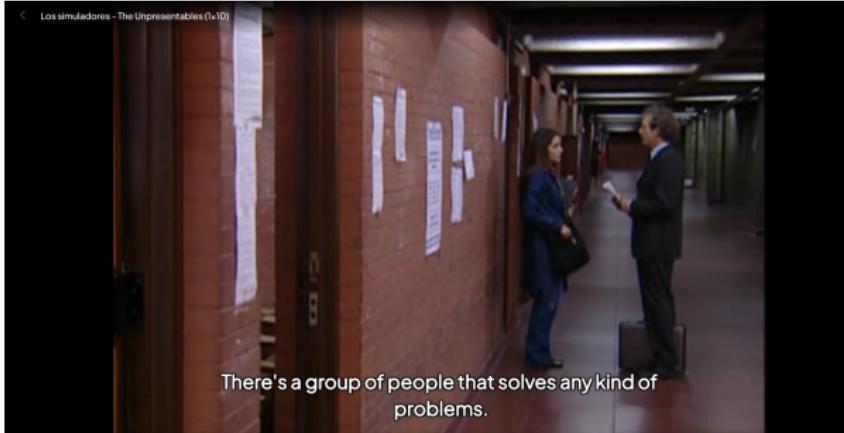
Algoritmos y Estructuras de Datos III (COMP930005)

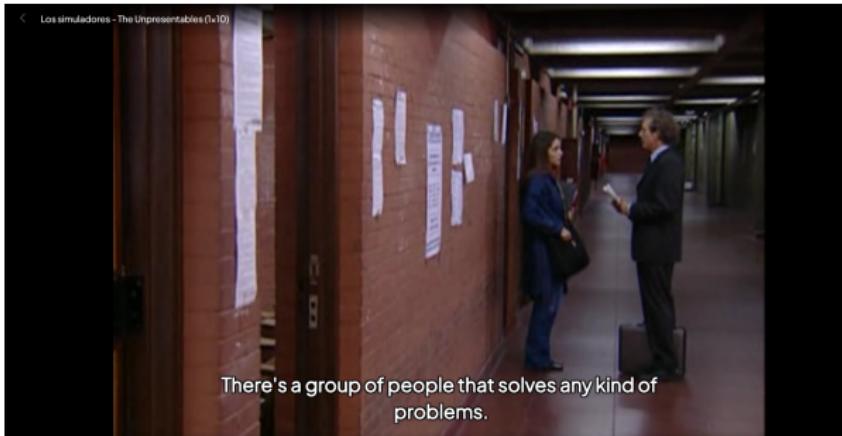
Comisión	Subcomisión	Ubicación	Responsabilidad	Turno	Inscripciones
Única	Teórico-Práctica (Teórico-Práctica)	Ciudad Universitaria	Ayudante de 2da	Sin definir	125

²<https://campus.exactas.uba.ar/course/view.php?id=1465§ion=2>

Algo III 2019 vs TM del 2024

- ▶ 2019
 - ▶ Estudiantes: 125
 - ▶ Docentes auxiliares: 8
 - ▶ Estudiantes por docente: 15,625
- ▶ 2024
 - ▶ Estudiantes: 190
 - ▶ Docentes auxiliares: 5
 - ▶ Estudiantes por docente: 38





There's a group of people that solves any kind of problems.



We will talk about the essay.



What happens in Germany, and all Europe at that time...



Espacio Publicitario: Deportes

Vení a hacer **DEPORTES**
en Exactas

Actividades recreativas

Básquet | Circuit Training |
Gimnasia artística | Karate |
Aikido | Pilates | Voley | Yoga

Actividades Interfacultades

Ajedrez | Básquet | Fútbol 11 |
Futsal | Handball | Hockey |
Natación | Tenis | Tenis de
Mesa | Voley



Todas las actividades son
libres y gratuitas. El apto
físico es obligatorio.

.UBA EXACTAS

Deportes. SECCB

<https://www.instagram.com/torneosinternosexactas/>
<https://www.instagram.com/deportesexactasuba/>

Fin del Espacio Publicitario

Travesía Vital

Hay un terreno, que podemos pensar como una grilla de m filas y n columnas, con trampas y pociones. Queremos llegar de la esquina superior izquierda hasta la inferior derecha, y desde cada casilla sólo podemos movernos a la casilla de la derecha o a la de abajo. Cada casilla i,j tiene un número entero $A_{i,j}$ que nos modificará el nivel de vida sumándonos el número $A_{i,j}$ (si es negativo, nos va a restar $|A_{i,j}|$ de vida). Queremos saber el mínimo nivel de vida con el que debemos comenzar tal que haya un camino posible de modo que en todo momento nuestro nivel de vida sea al menos 1. Por ejemplo, si tenemos la grilla

$$A = \begin{bmatrix} -2 & -3 & 3 \\ -5 & -10 & 1 \\ 10 & 30 & -5 \end{bmatrix}$$

el mínimo nivel de vida con el que podemos comenzar es 7 porque podemos realizar el camino que va todo a la derecha y todo abajo.

Top-Down vs Bottom-Up

Ambos son similares. En general la elección entre uno u otro enfoque depende, más que nada, del gusto del programador. Igual enumeramos algunas diferencias:

- ▶ Top-Down usa recursión, Bottom-Up es iterativo.
- ▶ En Top-Down la recursión va resolviendo sólo los subproblemas que necesita. En Bottom-Up, computamos todos.
- ▶ Bottom-Up tiene una ligera dificultad agregada, tenemos que decidir en qué orden iteramos para resolver los subproblemas.
- ▶ A veces el bottom-up permite usar menos memoria. Pero perdiendo la oportunidad de reconstruir una solución.