

TEORÍA DE ALGORITMOS  
(75.29) CURSO BUCHWALD - GENENDER

# Trabajo Práctico 2

## Programación Dinámica

10 de octubre de 2023

Stephanie Izquierdo  
104196

Tomás Macía  
99248

## 1. Análisis del problema

### 1.1. Problema

Luego de haber analizado todos los rivales ahora Scaloni lo que necesita es definir un cronograma de entrenamiento y desea hacerlo para los próximos  $n$  días.

Es sabido que el entrenamiento del día  $i$  demanda una cantidad de esfuerzo  $e_i$ , y podemos decir que la ganancia de dicho entrenamiento es igual al esfuerzo.

El entrenamiento que corresponde al día  $i$  (así como su esfuerzo y ganancia) son inamovibles. Si la cantidad de energía que se tiene para un día  $i$  es  $j < e_i$ , entonces la ganancia que se obtiene en ese caso es  $j$ .

Además se debe tener en cuenta que la cantidad de energía que tienen disponible para cada día va disminuyendo a medida que pasan los entrenamientos  $s_1 > s_2 > \dots > s_n$ .

Existe también la posibilidad de descansar un día logrando así que la energía se renueve es decir que la energía vuelva a ser  $s_1$ .

### 1.2. Entendiendo el problema

Habiendo explicado el problema se plantea como objetivo definir los días que deben entrenarse y los días que convenga a descansar de tal forma de obtener la mayor ganancia posible.

La problemática reside en que para obtener la mayor ganancia hay que definir cuándo nos conviene descansar de forma tal que el reinicio de energía sea bien aprovechado para el día actual ya que el esfuerzo inicial es el mayor y este va decreciendo según el pasar de los días.

Observamos entonces que la energía del día no varía, pero sí el esfuerzo gracias a este reinicio.

Entonces debemos determinar qué día descansar teniendo en cuenta las distintas posibilidades de ganancia según si anteriormente descansó el equipo.

Por ejemplo se podría considerar entrenar todos los días seguidos pero en el caso en que el esfuerzo actual sea menor al inicial y la energía del día actual sea mayor igual a la inicial convendría descansar el día anterior para poder aprovechar al máximo esa energía.

	1	2	3
Energía	5	27	13
Esfuerzo	35	14	10

Cuadro 1: Ejemplo promedio

En este caso, el óptimo sería descansar el primer día y luego entrenar los siguientes. De esta forma, el primer día cuya ganancia por entrenar es 5, descansamos, y estamos con la energía máxima para el segundo día. Como los jugadores tendrán 35 de energía y un esfuerzo requerido de 27 para obtener 27 de ganancia, nos quedamos con el mínimo que es 27. A esta ganancia acumulada, se le suma la ganancia del 3er día, donde tenemos como máximo una ganancia de 13 y un  $s_i$  de 14, entonces vamos a acumular 13 más a la ganancia del día anterior, que resulta en un máximo de 41.

Si analizamos otros posibles planes de entrenamiento, por ejemplo uno donde los jugadores entrenan todos los días, la ganancia sería  $\min(5, 35) + \min(27, 14) + \min(13, 10) = 29$ . Si descansamos el segundo día, sería  $\min(5, 35) + \min(13, 35) = 18$ .

Algo importante a notar es que no tiene sentido descansar el último día, porque siempre ese último día va a sumar más ganancia a la última ganancia acumulada, entonces siempre me va a aportar algo extra.

### 1.3. Solución

Nosotros planteamos los siguientes casos como las combinaciones posibles que se puede tener entre los descansos y entrenamientos:

- Como es posible entrenar o no entrenar el primer día la fórmula para los dos primeros días resulta en la siguiente:  $\min(e_i, s_j)$  siendo  $j$  la representación del primer día o segundo.
- Si se entrena todos los días consecutivos se obtendrá la ganancia del día actual más la ganancia del día anterior:  $\min(e_i, s_i) + OPT(e_{i-1}, s_{i-1})$
- Entendemos también que es redundante plantear esfuerzos de día futuros para el día actual.
- Para el segundo día en adelante, se debe tener en cuenta la ganancia máxima de el antepenúltimo día para decidir si se descansa o no el día anterior, con lo cual para este caso consideramos el mínimo entre el esfuerzo actual más la ganancia del óptimo de dos días atrás ya que esto nos indica si conviene descansar para reponer energía o seguir entrenando días consecutivos:  $\min(s_j, e_i) + \max(OPT(s_k, e_{i-2}))$  para todo  $kentre0yn$

Para ir armando la matriz de óptimos, nos quedamos con el máximo de las siguientes situaciones, presentando así la fórmula de recurrencia:

- Para el caso en el que se entrena el día anterior:  $\min(e_i, s_j) + OPT(s_{j-1}, e_{j-1})$
- Para el caso en que se descansa el día anterior:  $\min(e_i, s_j) + \max(OPT(s_k, e_{i-2}))$  para todo  $kentre0yn$

## 2. Algoritmo

En la sección anterior fue descrito el algoritmo en términos de pseudocódigo y los pasos a seguir para ir obteniendo los óptimos. A continuación se presenta el código del algoritmo desarrollado.

```
1 def scaloni_ganancia(e_i, s_i):
2     cant_dias = len(e_i)
3     matriz = np.zeros((cant_dias, cant_dias), dtype=int)
4
5     for columna in range(len(matriz[0])):
6         for fila in range(len(matriz)):
7             if fila == 0:
8                 if columna==0 or columna==1:
9                     matriz[fila][columna] = min(s_i[fila], e_i[columna])
10                else:
11                    matriz[fila][columna] = min(s_i[fila], e_i[columna]) + max(matriz[:,
12                    columna-2])
13                elif fila > columna: continue
14                else:
15                    matriz[fila][columna] = min(s_i[fila], e_i[columna]) + matriz[fila-1,
16                    columna-1]
17     return matriz
```

36	2	99	21	158
0	38	63	118	80
0	0	87	82	167
0	0	0	106	123
0	0	0	0	146

Cuadro 2: Ejemplo de 5 días

El algoritmo para obtener la ganancia máxima obtenida por la construcción de la matriz anterior es el siguiente:

```
1 def scaloni_plan(opt):
2     cant_dias = opt.shape[1]
3     idx_fila, idx_col = (np.argmax(opt[:, -1]), cant_dias - 1)
4     plan = [ENTRENO]
5
6     while len(plan) != cant_dias:
7         if idx_fila != 0:
8             idx_fila -= 1
9             idx_col -= 1
10            plan.append(ENTRENO)
11        else:
12            idx_fila = np.argmax(opt[:, idx_col-1]) # check if idx_col == 0
13            idx_col -= 1
14            plan.append(DESCANSO)
15
16     return plan[::-1]
```

En este algoritmo, comenzamos analizando la ganancia máxima correspondiente al ultimo día, ya que sabemos que el ultimo día si o si se entrena, y de esta forma vamos recorriendo la diagonal de manera inversa y en el caso en el que lleguemos al esfuerzo inicial, identificamos que el día anterior hubo un descanso y volvemos a realizar lo mismo hasta llegar al primer día. Por ultimo como este análisis fue iniciado del ultimo día al primero, devolvemos el arreglo realizado al revés.

<b>36</b>	2	<b>99</b>	21	158
0	38	63	<b>118</b>	80
0	0	87	82	<b>167</b>
0	0	0	106	123
0	0	0	0	146

Cuadro 3: Ejemplo de planificación a 5 días

ENTRENO    DESCANSO    ENTRENO    ENTRENO    ENTRENO

Cuadro 4: Resultado de planificación a 5 días

### 2.0.1. Complejidad

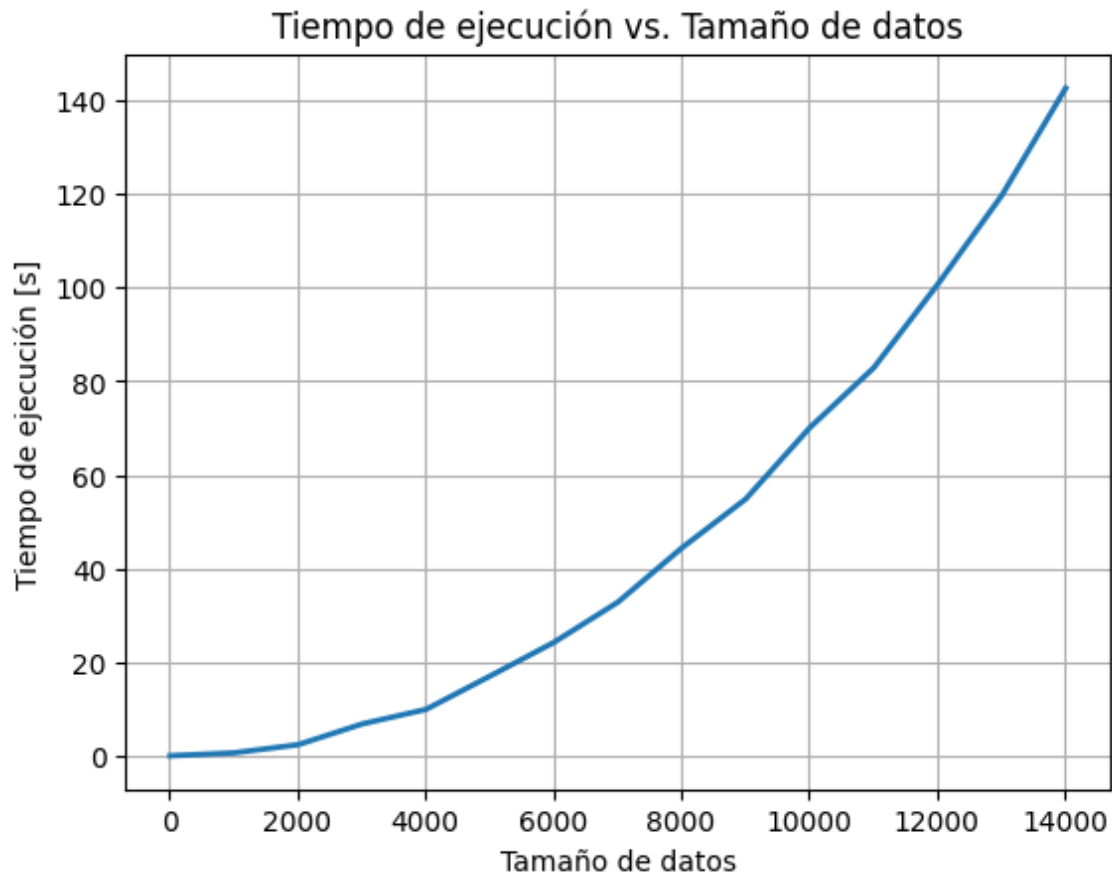
La complejidad del algoritmo propuesto para encontrar el orden optimo de entrenamiento es  $\mathcal{O}(n^3)$ , debido a que necesitamos generar una matriz de tamaño  $n \times n$  siendo  $n$  la cantidad de días y luego para cada posición se realizan operaciones de mínimo y máximo siendo la de mínimo  $\mathcal{O}(1)$  ya que comparamos 2 valores y la de máximo  $\mathcal{O}(n)$  ya que comparamos toda una fila. Dando así una complejidad de  $\mathcal{T}(n^2 * n)$

Pero además como la matriz anterior resulta triangular debido a las aclaraciones hechas en el análisis obtenemos como resultado  $\mathcal{T}(n^2 * n/2)$  Lo cual resulta en  $\mathcal{O}(n^3)$

La complejidad de la obtención del plan es  $\mathcal{O}(n)$  ya que vamos recorriendo las columnas de la matriz buscando el máximo y realizando operaciones  $\mathcal{O}(1)$

## 3. Mediciones

Se realizaron mediciones en base a crear distintas combinaciones de  $e_i$  y  $s_i$ , cuyos valores se encuentran entre 1 y 100. Se ejecutó el algoritmo con distintos valores de  $n$ , es decir el tamaño de los datos (días de entrenamiento). Estos datos van desde 10 elementos hasta aproximadamente 15000 elementos, con un intervalo de 100. Es decir se calculó con 10, 1010, 2010.... 14010 elementos y se midió el tiempo que tardó únicamente la ejecución del algoritmo (el tiempo de generación del set de pruebas random no fue medido). La elección de estos valores y saltos fueron elegidos en base a prueba y error y a los tiempos que nos demoró en realizar ejecuciones con distintos tamaños de datos



#### 4. Conclusiones

En síntesis, el enfoque desarrollado en este trabajo aborda eficazmente el desafío de optimizar la ganancia de análisis de días de entrenamiento del equipo de Lionel Scaloni. Se ha identificado que el factor crítico es determinar los días de descanso para que los jugadores obtengan mejores resultados en los días convenientes. A través de un algoritmo programación dinámica, se logra asignar la ganancia de manera estratégica teniendo en cuenta el reinicio de esfuerzo. Esta solución ofrece una forma efectiva y eficiente de abordar el problema, permitiendo a Scaloni entrenar de manera óptima a su equipo.