Back-track to the EDA

Equipo Sort

Al final no voy a llegar al máximo de gigavatios en cada tema de EDA. ¡No seré un McFly experto! Si lo hubiera sabido, habría hecho más misiones de los temas en los que me faltan y menos en los que me sobran. Claro, que también podría viajar al pasado y avisar a mi yo de septiembre de las misiones mínimas que tengo que hacer para conseguir el 10.

Además de las misiones estándar por cada tema, tenemos unas misiones extra que se pueden asignar a algún tema específico que vaya flojo de puntuación: las *misiones a la carta*.

No puedo permitirme hacer más misiones de las necesarias, porque tengo 6 asignaturas más y tengo que sacar tiempo de debajo de las piedras. Por otro lado, supongo que tardo siempre lo mismo en hacer cada misión: al final siempre acabo dedicando una tarde entera. Por eso me interesa hacer el mínimo número de misiones. También voy a suponer que voy a conseguir todos los gigavatios de las misiones que haga; ¡les voy a dedicar muchísimo tiempo!

La profesora es muy buena y siempre da misiones suficientes como para llegar a ese 10. Otra cosa es que nos cueste sudor y lágrimas (o una segunda matrícula en Cálculo Diferencial).

Entrada

La entrada estará compuesta por distintos casos de prueba. En la primera línea aparecen tres números: el número de gigavatios que necesito alcanzar en cada tema ($0 < G \leq 10^9$), el número $T \in \mathbb{N}$ de temas en el cuatrimestre y el número $M \in \mathbb{N}$ de misiones por cada tema ($T * M \leq 20$).

Por cada línea de las T siguientes líneas se encuentran M enteros, que corresponden a los gigavatios asignados a cada misión de dicho tema. Una misión puede no estar asignada en algún tema, en cuyo caso se me concedería cero gigavatios por hacerla. Además ninguna misión resta gigavatios, ¡encima de hacer el esfuerzo!

Tras esto, una última línea contiene M enteros, que corresponden a los gigavatios asignados a *misiones a la carta*.

La entrada termina con una línea con tres ceros que no debe procesarse.

Salida

Por cada caso de prueba se escribirá el número mínimo de misiones que tengo que realizar para llegar a G gigavatios en cada tema.

Entrada de ejemplo

```
1 1 1
1
0
4 2 2
1 1
3 1
2 0
20 4 5
4 8 2 2 4
4 8 2 0 6
4 14 2 0 4
4 8 2 0 6
0 0 0 0 0
20 4 5
4 8 2 2 4
4 8 2 0 6
4 14 2 0 4
4 8 2 0 6
5 0 0 0 0
4 4 5
1 1 1 1 1
1 1 1 1 1
1 1 1 1 1
1 1 1 1 1
3 2 1 1 0
0 0 0
```

Salida de ejemplo

```
1
5
16
15
13
```

Índice

Solución de búsqueda exhaustiva

En manostijeras.cpp se guardan en `gigasxtema` los gigavatios acumulados por cada tema. Se ordenan decrecientemente las filas de la matriz.

Por cada tema (`i`), se llama a la función `rec` de manera acumulativa con las misiones: ninguna, la primera, la primera y la segunda... (tras haberlas incluido en `gigasxtema` y haber incrementado `solparcial`).

Si hemos acabado la matriz de misiones estándar, quedan por añadir las posibles *misiones a la carta*. Entonces se llama a `final`.

En `final` probamos todas las opciones añadiendo o no añadiendo cada *misión a la carta* a cada uno de los temas mediante el bucle.

Si hemos acabado con todas las misiones, se comprueba si el estado actual es solución (si la solución parcial es menor que la total y si en todos los temas se ha llegado al máximo `g`).

Poda 1: Podar los ceros

En manostijeras1.cpp se podan todos los nodos no positivos, ya que solo añaden volumen de misiones sin aportar gigavatios. Se podan tanto las misiones estándar como las *a la carta*.

Poda 2: Podar los que se pasan de gigavatios

En manostijeras2.cpp se podan (además de los anteriores) todos los nodos que sobrepasen el número de gigavatios requerido `g`, ya que una vez se pasa de `g` en un tema no hacen falta más misiones. Se podan tanto las misiones estándar como las *a la carta*.

Poda 3: Podar los peores que lo que ya tenemos

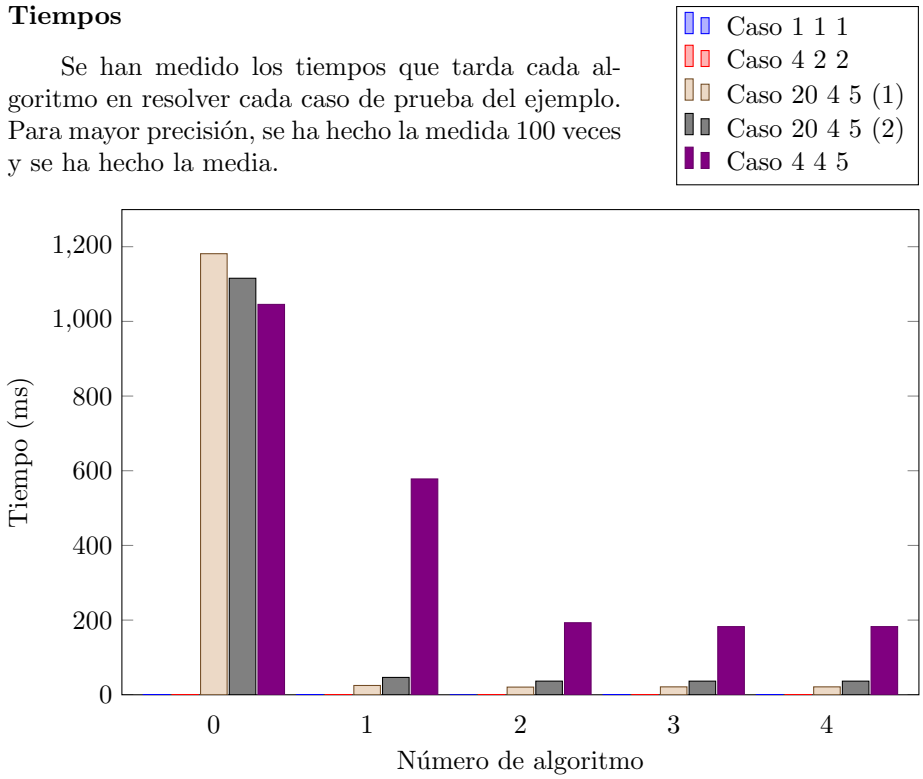
En manostijeras3.cpp se podan (además de los anteriores) todos los nodos en los que la solución parcial (número de misiones) no sea inferior a la solución total, ya que a partir de ese punto solo se van a añadir más misiones y ya tenemos una solución igual o mejor. Se podan tanto las misiones estándar como las *a la carta*.

Poda 4: Podar los peores que la forma polinómica

En manostijeras4.cpp acotamos mejor la solución total: nos damos cuenta de que hay una manera de resolverlo de manera polinómica si no existieran las *misiones a la carta*, por lo que, si hay solución polinómica, nuestra solución mínima será menor o igual a ésta. Por tanto, se inicializa `sol` a este resultado y, en su defecto, al límite superior de los enteros (lo que había antes).

Tiempos

Se han medido los tiempos que tarda cada algoritmo en resolver cada caso de prueba del ejemplo. Para mayor precisión, se ha hecho la medida 100 veces y se ha hecho la media.

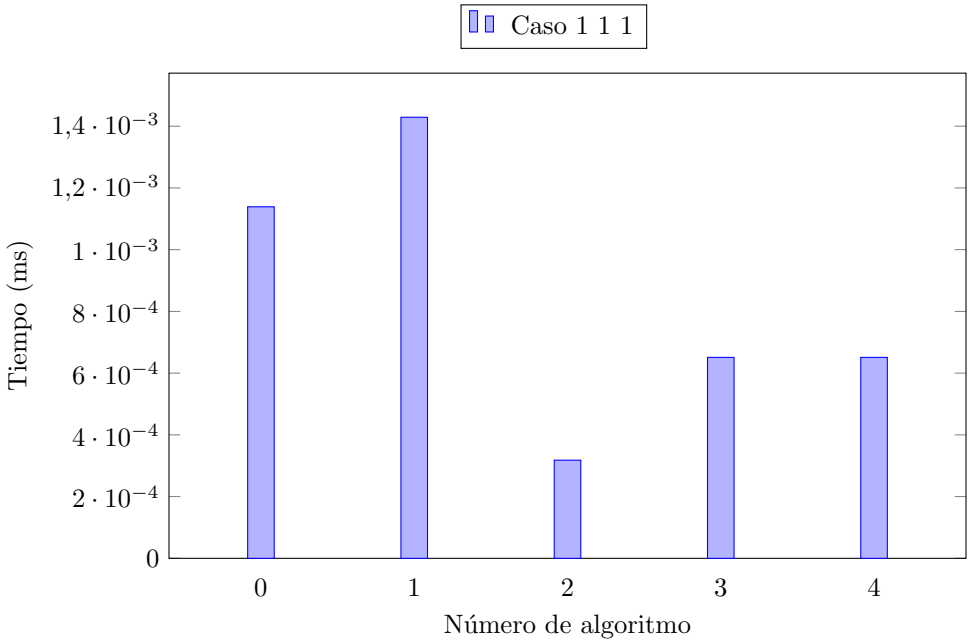


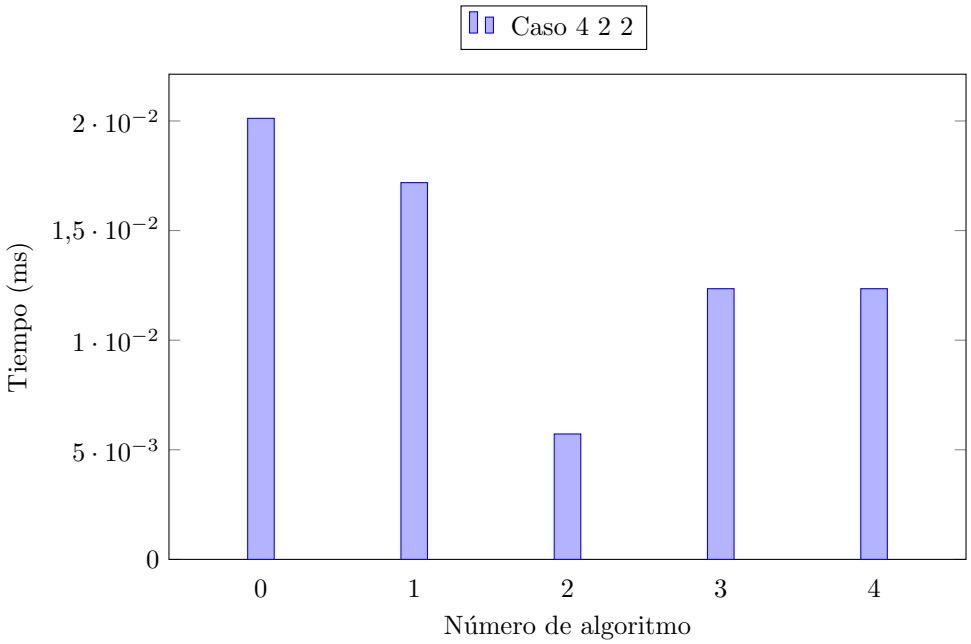
Como conclusión, la primera poda reduce significativamente el tiempo de los casos que tienen más ceros (por ello, el último caso no se reduce tanto). La segunda poda es la que parece mejor, ya que no hay casos que dependan tanto del número de soluciones posibles (que es el objetivo de las podas 3 y 4), y estas dos últimas podas compensan lo que se ahorra con lo que se gasta. En general, no hay gran diferencia entre las tres últimas podas.

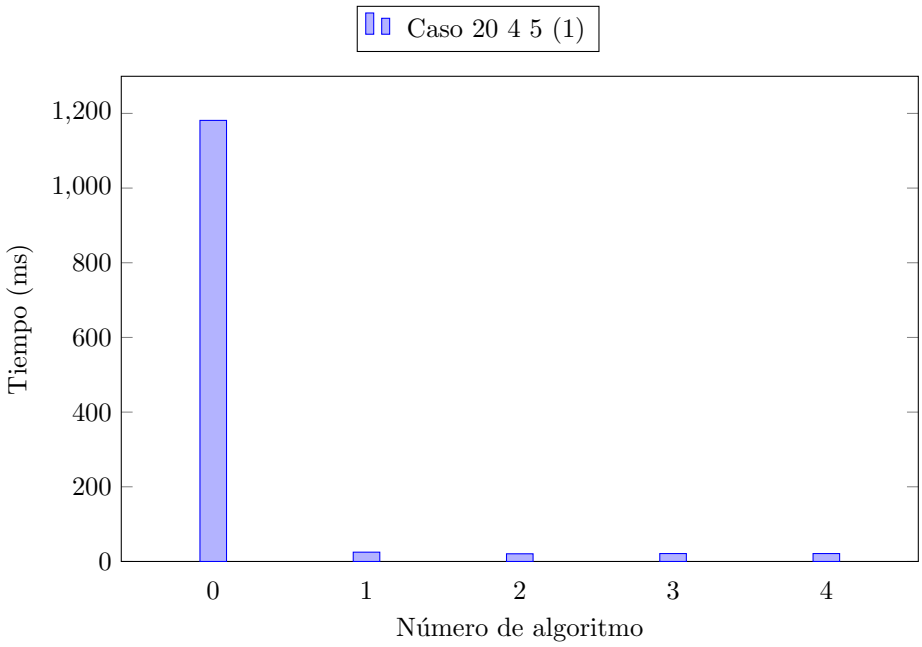
Cuanto más grande es el caso, más se nota la mejora de la poda. En los casos pequeños apenas se nota, mientras que en los casos más grandes es casi imposible ejecutar los primeros algoritmos.

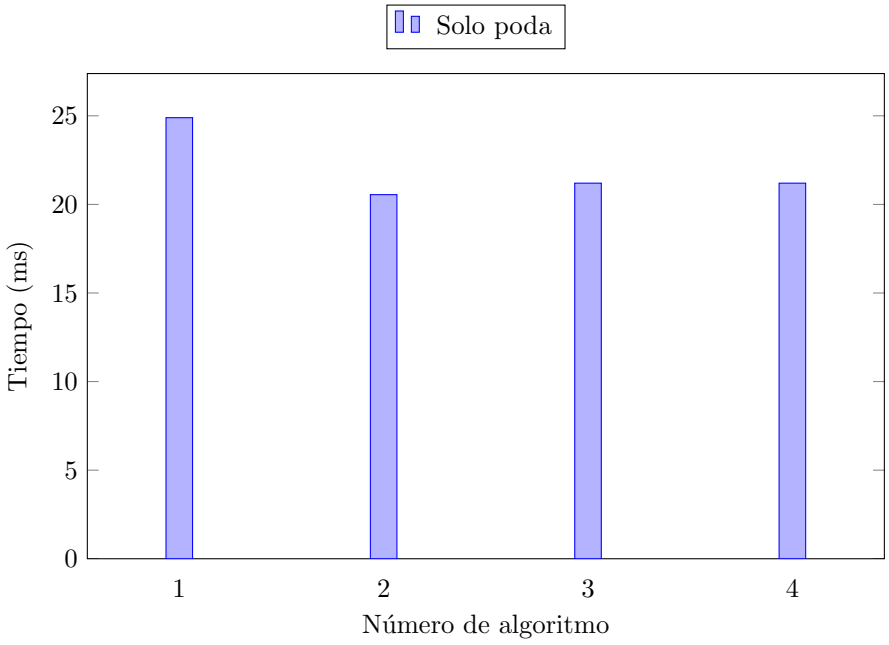
Código cronómetro: tiempos.cpp
Código de los algoritmos: algoritmos.cpp, algoritmos.h
Texto de entrada: salida.txt
Textos de salida: salida.txt, salida1.txt, salida2.txt, salida3.txt, salida4.txt.

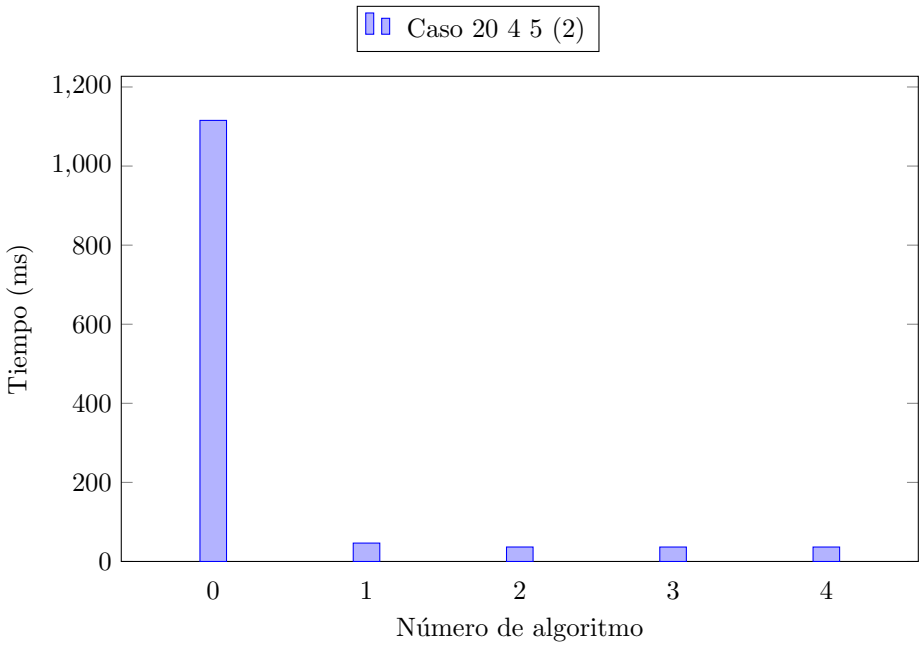
Gráficas de tiempos

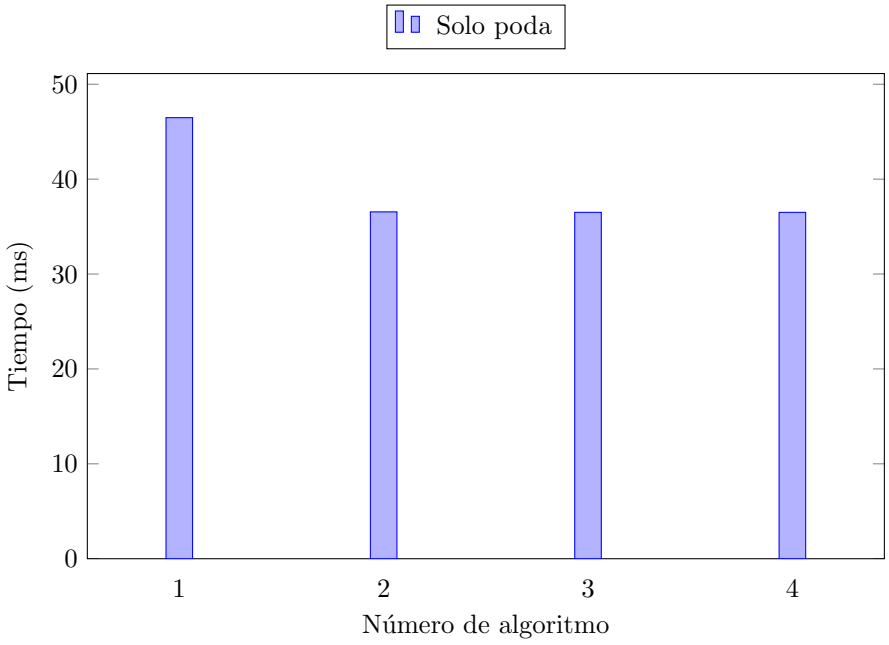





Caso 20 4 5 (1)
1,200
1,000
800
600
400
200
0
Tiempo (ms)
0
1
2
3
4
Número de algoritmo


Solo poda
25
20
15
10
5
0
Tiempo (ms)
1
2
3
4
Número de algoritmo


Caso 20 4 5 (2)
1,200
1,000
800
600
400
200
0
Tiempo (ms)
0
1
2
3
4
Número de algoritmo


Solo poda
50
40
30
20
10
0
Tiempo (ms)
1
2
3
4
Número de algoritmo

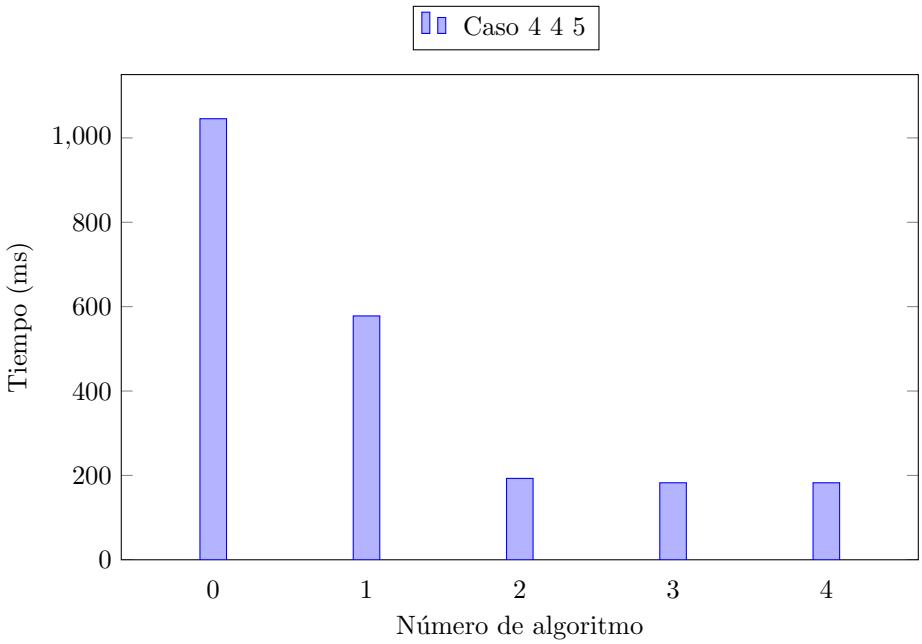


Tabla de tiempos

Algoritmo	C. 1 1 1	C. 4 2 2	C. 20 4 5 (1)	C. 20 4 5 (2)	C. 4 4 5
0	$1{,}14 \cdot 10^{-3}$	$2{,}01 \cdot 10^{-2}$	1181,28	1115,51	1045,51
1	$1{,}43 \cdot 10^{-3}$	$1{,}72 \cdot 10^{-2}$	24,9	46,48	578,04
2	$3{,}18 \cdot 10^{-4}$	$5{,}72 \cdot 10^{-3}$	20,55	36,55	192,91
3	$6{,}51 \cdot 10^{-4}$	$1{,}23 \cdot 10^{-2}$	21,2	36,5	182,53
4	$6{,}51 \cdot 10^{-4}$	$1{,}23 \cdot 10^{-2}$	21,2	36,5	182,53