Problema del corte de vástagos (Rod Cutting)

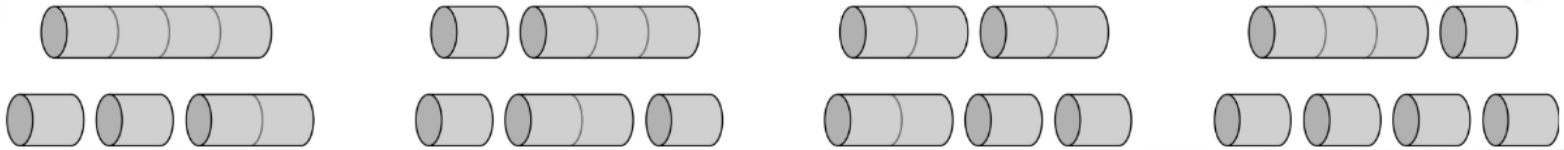
Grupo L1_7

- Pablo Pastor Martín
- Isaac Manuel Aimán Salas
- Javier Ramos Fernández

Introducción

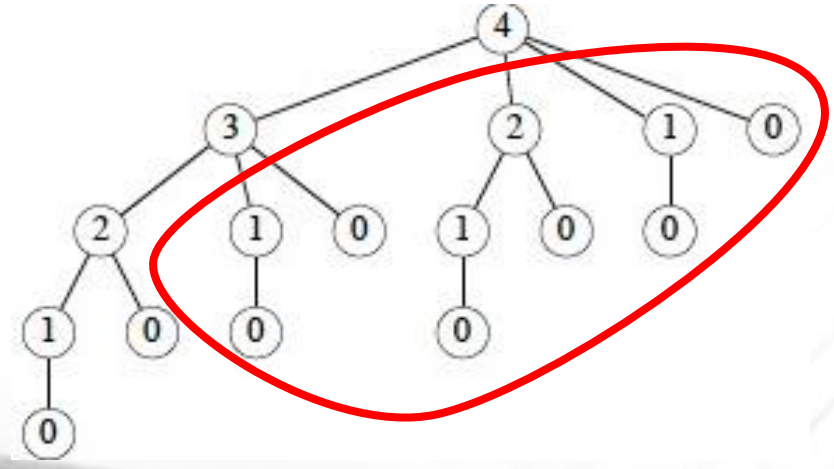
El problema del corte de vástagos busca, dada una relación de precios para fragmentos de un cierto tamaño, maximizar el precio resultante de la hipotética venta de los mismos.

- Es un problema típicamente relacionado con la programación dinámica.
- Es posible enfocarlo mediante Divide y Vencerás, pero su rendimiento es pobre.



Enfoque mediante Divide y Vencerás

- Se solución óptima se basa en la composición de una o varias divisiones de la barra.
- Complejidad: $T(n) = \sum_{k=0}^{n-1} T(j) = \left(\sum_{k=0}^{n-1} 2^j\right) + 1 = 2^n$
  - Es fácilmente demostrable mediante inducción.
- Por tanto, la complejidad es exponencial, lo que hace que este enfoque no sea el adecuado para resolver este problema.
- Esto es debido a que se hacen llamadas innecesarias.



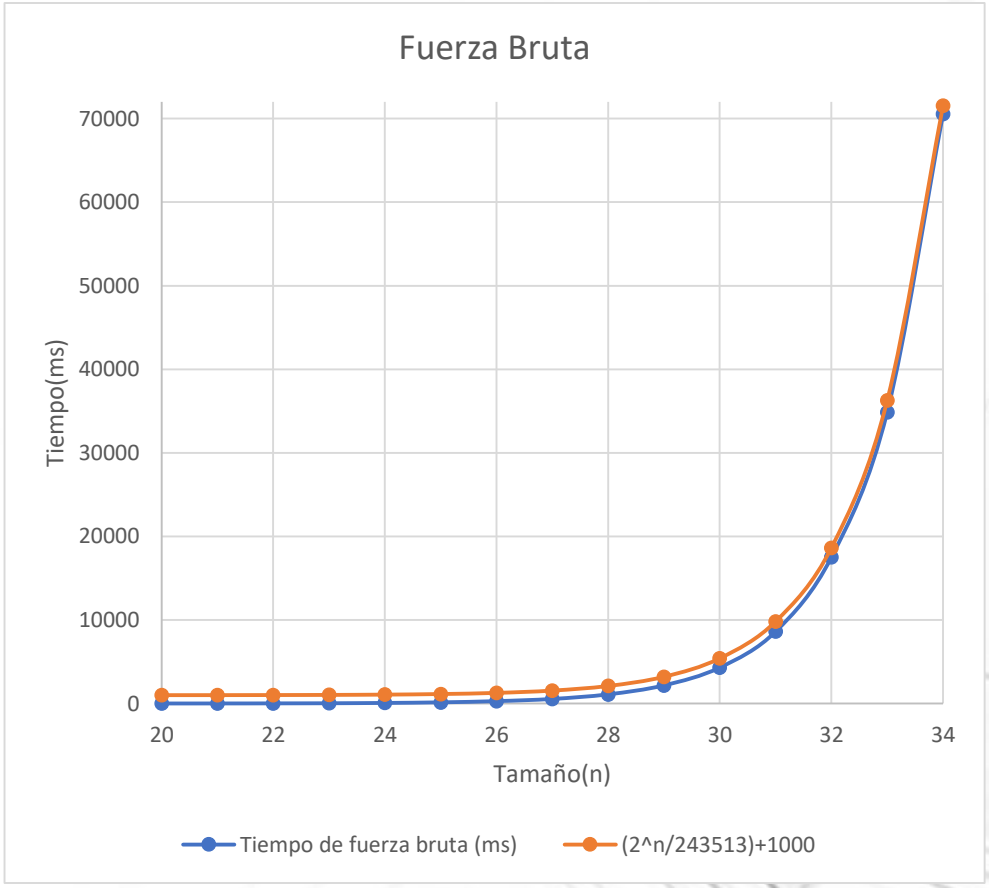
Enfoque mediante Divide y Vencerás

- **Pseudocódigo:**

```
CUT-ROD(p, n)
1    if (n == 0)
2    return 0
3    q = -inf
4    for i = 1 to n
5         q = max(q, p[i] + CUT-ROD(p, n - i))
6    return q
```

Enfoque mediante Divide y Vencerás

Tamaño de entrada	Tiempo de ejecución
23	34
25	143
27	551
29	2175
31	8610
33	34874
34	70550



Enfoque mediante Programación Dinámica (Bottom Up)

- Se resuelven los problemas empezando por longitudes de cuerda pequeñas.
- En un vector de tamaño n + 1, guardamos los beneficios óptimos para cada longitud de corte.
- Complejidad: $\sum_{i=1}^{n}\sum_{j=1}^{i} c = \sum_{i=1}^{n} jc = c\frac{(1+n)n}{2} = \Theta(n^2)$

n	i	j	Total
1	1	1	2
2	2	3	5
3	3	6	9
4	4	10	14
5	5	15	20

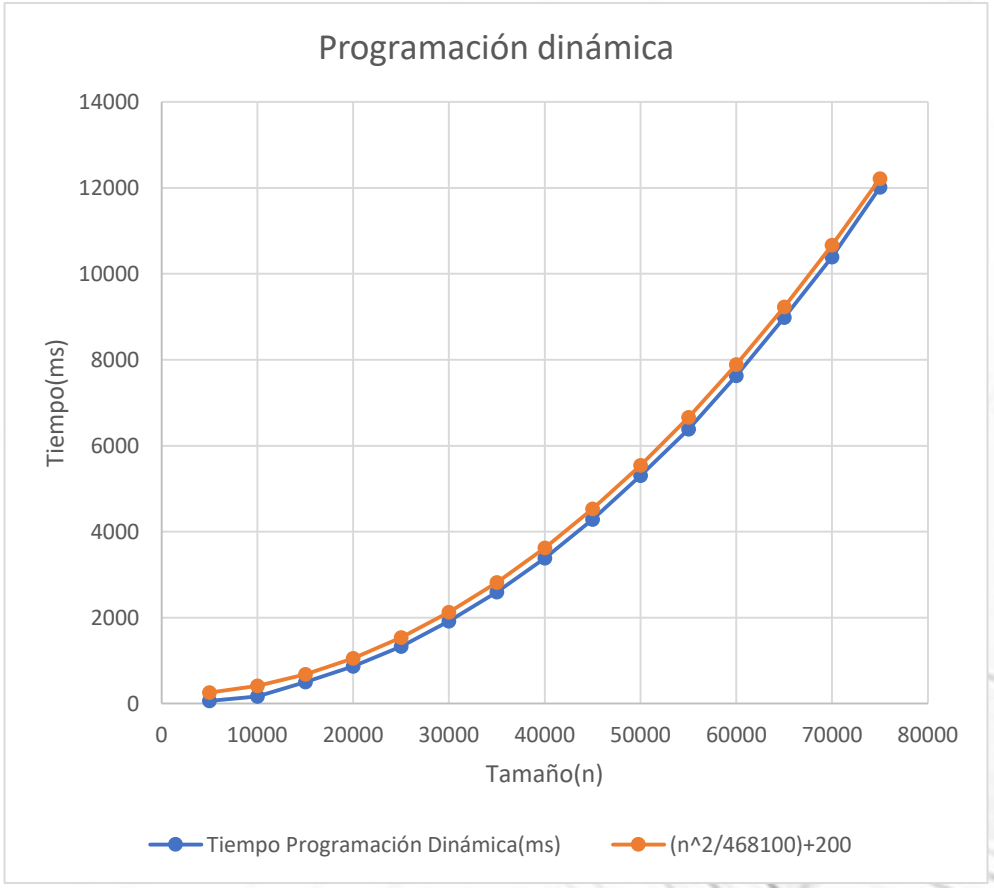
Enfoque mediante Programación Dinámica

- **Pseudocódigo:**

```
EXTENDED-BOTTOM-UP-CUT-ROD(p, n)
1    Sean r[0..n] y s[0..n] nuevos arrays
2    r[0] = 0
3    for j = 1 to n
4         q = -inf
5         for i = 1 to j
6              if q < p[i] + r[j - i]
7                   q = p[i] + r[j - i]
8                   s[j] = i
9         r[j] = q
10   return (r, s)
```

Enfoque mediante Programación Dinámica (Bottom Up)

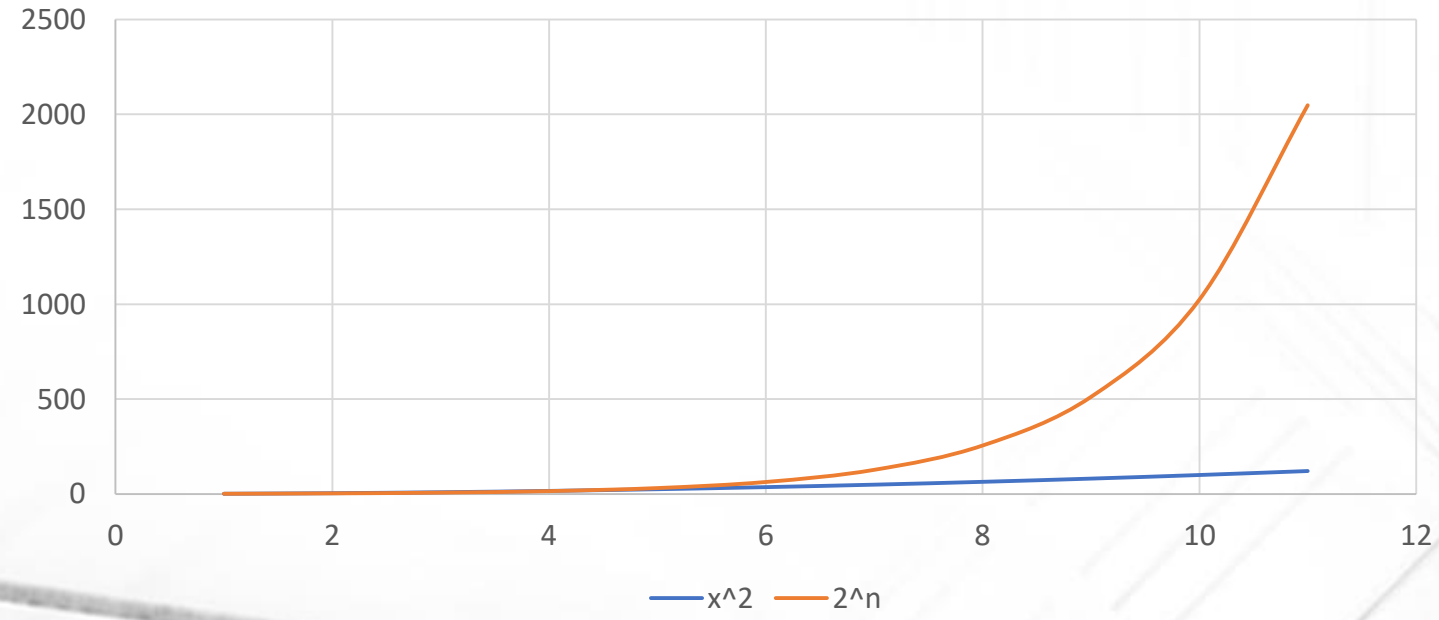
Tamaño de entrada	Tiempo de ejecución
10000	168
20000	869
30000	1918
40000	3381
50000	5305
60000	7625
70000	10389



Resumiendo

- La diferencia entre los tiempos de ejecución es abismal
- Podemos concluir que la programación dinámica es el enfoque más adecuado para resolver el problema

Tiempo(ms)	Tamaño para programación dinámica	Tamaño para divide y vencerás
1100	24000	28
8600	64000	31

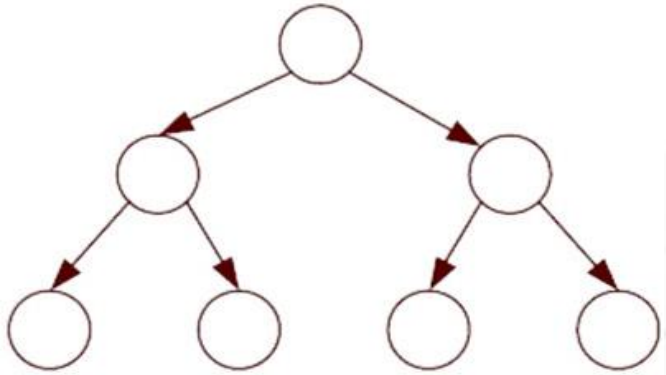


Conclusiones

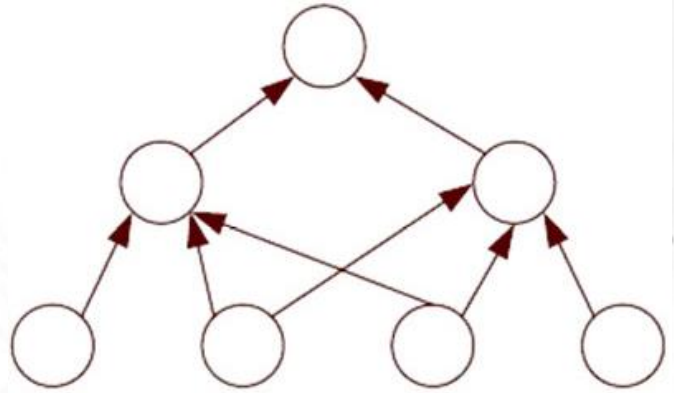
- La eficiencia de un algoritmo es una característica importantísima del mismo, puesto que permitirá trabajar con mayores volúmenes de datos requiriendo un menor tiempo.

- En esta asignatura hemos aprendido que el paradigma Divide y Vencerás es tremendamente eficiente en ciertos casos (MergeSort)

- Sin embargo, en ocasiones como este problema, este paradigma se ve ampliamente superado por otros como la Programación Dinámica



Divide y vencerás



Programación dinámica

¿Preguntas?

Bibliografía

- *Proving number of calls made in cutrod algorithm.* (2017). *Cs.stackexchange.com*. Revisado el 19 maezo de 2017, desde http://cs.stackexchange.com/questions/29661/provingnumberofcalls-madeincutrodalgorithm

- Pecelli, P. (2009). *Analysis of Algorithms Dynamic Programming for Rod Cutting.* Revisado el 19 de marzo de 2017, desde http://www.cs.uml.edu/~kdaniels/courses/ALG_503_F12/DynamicRodCutting.pdf

- *Lecture 12: Dynamic Programming Rod Cutting.* (2017). *Faculty.ycp.edu*. Revisado el de 19 marzo de 2017, desde http://faculty.ycp.edu/~dbabcock/PastCourses/cs360/lectures/lecture12.html

- Kretchmar, R. (2017). *Rod Cutting: Example DP Solution* (1st ed.). Revisado el 19 de marzo de 2017, desde http://personal.denison.edu/~kretchmar/271/DPRodCuttingExample.pdf