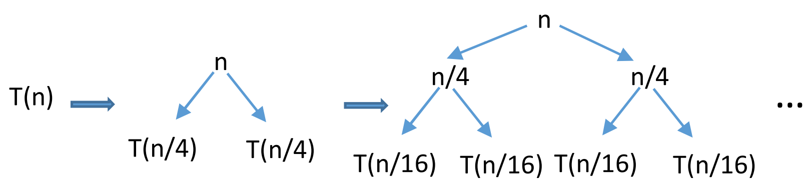
Ders 9:

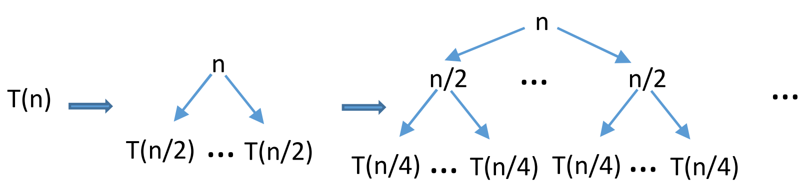
- T(n)=2T(n/4)+n, bunun recursion tree'sini çizelim.



1 adet n → n
2 adet n/4 → n/2
4 adet n/16 → n/4
8 adet n/64 → n/8
.....

2^h adet 1

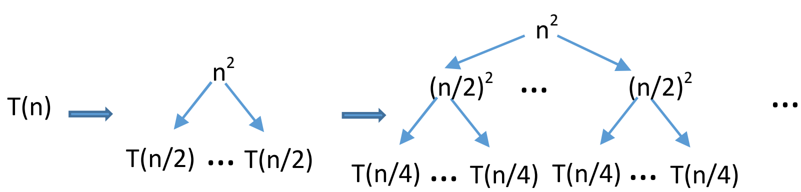
  - h=$\log_4 n$
  - $T(n) = n + \frac{n}{2} + \frac{n}{4} + \frac{n}{8} + \cdots = n\left(1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \cdots\right) = n\sum_{i=0}^{log_4 n} (\frac{1}{2})^i$
  - $T(n) = n\frac{\frac{1}{2}^{(log_4 n)+1} - 1}{\frac{1}{2} - 1} = n\frac{1 - \frac{1}{2}^{(log_4 n)+1}}{\frac{1}{2}} = 2n\left(1 - \frac{1}{2}^{(log_4 n)+1}\right) = 2n\left(1 - 2^{-(log_4 n)-1}\right)$
  - $T(n) = 2n\left(1 - \frac{2^{-(log_4 n)}}{2}\right) = 2n\left(1 - \frac{n^{-(log_4 2)}}{2}\right) = 2n\left(1 - \frac{n^{-2)}}{2}\right) = 2n\left(1 - \frac{1}{2\sqrt{n}}\right) =$ $2n - \sqrt{n} = O(n)$
- T(n)=8T(n/2)+n için recursion tree'i çiziniz.



1 adet n
8 adet n/2
64 adet n/4
512 adet n/8
.....

  - $T(n) = n + 4n + 16n + 64n + \cdots = n(1 + 4 + 16 + 64 + \cdots) = n\sum_{i=0}^{h} 4^i$
  - $h = log_2 n, T(n) = n\sum_{i=0}^{log_2 n} 4^i = n\frac{4^{(log_2 n)+1} - 1}{4-1} = \frac{n}{3}(4 * 4^{log_2 n} - 1)$
  - $T(n) = \frac{n}{3}\left(4n^{log_2 4} - 1\right) = \frac{n}{3}(4n^2 - 1) = O(n^3)$

- T(n)=8T(n/2)+n^2 için recursion tree'i çiziniz.

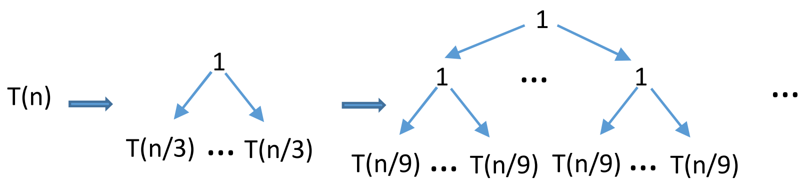


1 adet n^2
8 adet $(n/2)^2$
64 adet $(n/4)^2$
512 adet $(n/8)^2$
.....

  - $T(n) = n^2 + 2n^2 + 4n^2 + 8n^2 + \cdots = n^2(1 + 2 + 4 + 8 + \cdots) = n^2\sum_{i=0}^{h} 2^i$
  - $h = log_2 n, T(n) = n^2\sum_{i=0}^{log_2 n} 2^i = n^2\frac{2^{(log_2 n)+1} - 1}{2-1} = n^2(2 * 2^{log_2 n} - 1)$
  - $T(n) = n^2\left(2n^{log_2 2} - 1\right) = n^2(2n - 1) = O(n^3)$

- T(n)=8T(n/2)+1 için O(n) ?
  - T(n)=O(n^3)

- T(n)=4T(n/3)+1 için recursion tree'i çiziniz.

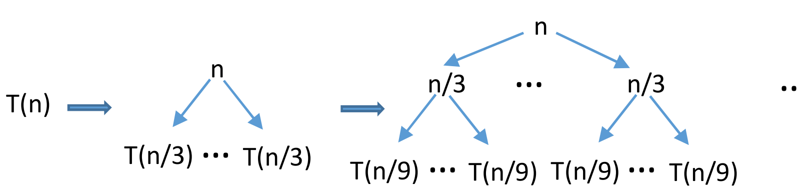


1 adet 1
4 adet 1
16 adet 1
64 adet 1
.....

  - $T(n) = \sum_{i=0}^{h} 4^i,\ h = log_3 n$
  - $T(n) = \sum_{i=0}^{log_3 n} 4^i = \frac{4^{(log_3 n)+1}-1}{4-1} = \frac{4*4^{log_3 n}-1}{3} = \frac{4*n^{log_3 4}-1}{3} = O(n^{log_3 4})$

- T(n)=9T(n/3)+1 için recursion tree'i çiziniz.
  - $T(n) = \sum_{i=0}^{h} 9^i,\ h = log_3 n$
  - $T(n) = \sum_{i=0}^{log_3 n} 9^i = \frac{9^{(log_3 n)+1}-1}{9-1} = \frac{9*9^{log_3 n}-1}{8} = \frac{9n^{log_3 9}-1}{8} = \frac{9*n^2-1}{8} = O(n^2)$

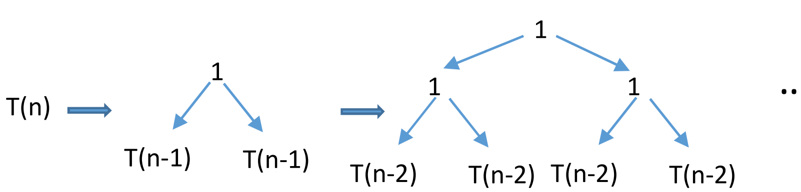
- T(n)=4T(n/3)+n için recursion tree'i çiziniz.



1 adet n
4 adet n/3
16 adet n/9
64 adet n/27
.....

  - $T(n) = n + 4n/3 + 16n/9 + 64n/27 + \cdots = n(1 + 4/3 + 16/9 + 64/27 + \cdots) = n\sum_{i=0}^{h}(\frac{4}{3})^i,\ h = log_3 n, T(n) = n\sum_{i=0}^{log_3 n}(\frac{4}{3})^i$
  - $\sum_{i=0}^{log_3 n}(\frac{4}{3})^i = \frac{(\frac{4}{3})^{(log_3 n)+1}-1}{\frac{4}{3}-1} = 3*\left(\frac{4}{3}\left(\frac{4}{3}\right)^{log_3 n} - 1\right) = 4\left(\frac{4}{3}\right)^{log_3 n} - 3 = 4n^{log_3 \frac{4}{3}} - 3$
  - $= 4n^{log_3(\frac{1}{3}*4)} - 3 = 4n^{log_3\frac{1}{3}+log_3 4} - 3 = 4n^{(log_3 4)-1} - 3$
  - $T(n) = n(4n^{(log_3 4)-1} - 3) = 4n^{log_3 4} - 3n = O(n^{log_3 4})$

- T(n)=2T(n-1)+1 için recursion tree'i çiziniz.



1 adet 1
2 adet 1
4 adet 1
8 adet 1
.....

  - $T(n) = 1 + 2 + 4 + 8 + 16 + \cdots = \sum_{i=0}^{h} 2^i,\ h = n$
  - $T(n) = \sum_{i=0}^{n} 2^i = \frac{2^{n+1}-1}{2-1} = O(2^n)$, çok fazla
- T(n)=T(n-1)+T(n-2)+1 olursa, fibonacci
  - $T(n) = O((\frac{1+\sqrt{5}}{2})^n)$, bu da çok fazla

  - ($\frac{1+\sqrt{5}}{2}$) golden ratio ≈ 1.618, 2^n'den küçük çünkü bazı recursion'lar daha önce bitiyor. Hepsi aynı seviyede değil. Önceden fibonacci(n)'i O(n) ile hesaplayabiliyorduk. Bu şekilde (fib(n)=fib(n-1)+fib(n-2)) recursive olarak tanımlarsak çok daha kötü oldu. ☹
- Fibonacci(n)'i O(n)'den daha iyi bulabilir miyiz?
  - $F = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$ olsun. $F^2 = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} * \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 2 & 1 \\ 1 & 0 \end{bmatrix}$
  - $F^3 = \begin{bmatrix} 2 & 1 \\ 1 & 0 \end{bmatrix} * \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 3 & 2 \\ 2 & 1 \end{bmatrix}, F^4 = \begin{bmatrix} 3 & 2 \\ 2 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 5 & 3 \\ 3 & 2 \end{bmatrix},$
  - $F^5 = \begin{bmatrix} 5 & 3 \\ 3 & 2 \end{bmatrix} * \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 8 & 5 \\ 5 & 3 \end{bmatrix}$
  - $F^n = \begin{bmatrix} fib(n+1) & fib(n) \\ fib(n) & fib(n-1) \end{bmatrix}$ güzel ☺
  - F^n'i lineer olarak hesaplarsak karmaşıklık yine O(n) olur. Ama bir sayının üssünü $\log_2 n$'le bulabilmiştik. Aynı şekilde bir matrisinde üssünü $\log_2 n$'de bulabilir miyiz?
  - hpower'ın aynısı sadece T=k*k'da k'lar sayı değil, 2*2'lik matris. 2*2'lik 2 matris sabit sayıda işlemle çarpılır. Yani n'e bağlı değil. Dolayısıyla F^n'de $\log_2 n$'le bulanabilir. İşte bu süper ☺