6 uždavinys:

Suraskime rekursinės lygties $T(n) = 2T\left(\frac{n}{3}\right) + n^2$sprendinį.

Pastaba: Kad nereikėtų perrašinėti medžio su kiekviena rekursijos iteracija galima ant šakos/lanko rekursinių iškvietimų įvertinimus, o mazguose rašyti likusią dalį rekurentinės lygties.

…
1–5 paveikslėlis

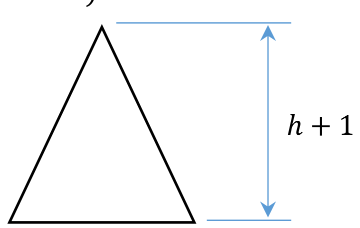
Mažiems $n \leq n_0$ programos sudėtingumas yra konstanta, t. y. $T(n) = O(1)$. Sprendžiamo uždavinio atveju $n_0 = 5$ ar $n_0 = 0$ Gautus medyje įvertinimus reikia susumuoti:

$$T(n) = \sum_{i=0}^{h} \left(\frac{2}{9}\right)^i n^2 = n^2 \sum_{i=0}^{h} \left(\frac{2}{9}\right)^i$$

čia h – medžio aukštis.
Įvertinkime, šią sumą iš viršaus, t. y. patį blogiausią programos, kurios darbo laiką/sudėtingumą aprašo sprendžiama rekurentinė lygtis, atvejį.
Kadangi po sumos ženklu stovi mažėjančios geometrinės progresijos nariai, nedaug pabloginsime įvertinimą jei sumuosime iki begalybės, t. y.

$$T(n) = n^2 \sum_{i=0}^{h} \left(\frac{2}{9}\right)^i < n^2 \sum_{i=0}^{\infty} \left(\frac{2}{9}\right)^i = \frac{n^2}{1 - \frac{2}{9}} = \frac{9}{7} n^2$$

$T(n) = O(n^2)$, nes įvertinimą didinome.
Įvertinant $T(n)$ geriausiu atveju reikės įvertinti h (kokio gylio rekursija). Kiekvieną kartą sprendžiamas uždavinys mažėja trigubai, bet šakų skaičius didėja trigubai. Visiems n medis bus pilnas, nepriklausomai ar lygtyje imsime $\left\lfloor \frac{n}{3} \right\rfloor$ ar $\left\lceil \frac{n}{3} \right\rceil$. Sudarytame medyje sumuojamų elementų skaičius bus $m = \sum_{i=0}^{h} 3^i = \frac{1}{2}\left(3^{h+1} - 1\right)$



Tokiu atveju $h = \lfloor \log_3 n \rfloor$ ir

$$T(n) = n^2 \sum_{i=0}^{h} \left(\frac{2}{9}\right)^i \geq n^2 \sum_{i=0}^{\lfloor \log_3 n \rfloor} \left(\frac{2}{9}\right)^i = n^2 \frac{\left(\frac{2}{9}\right)^{\lfloor \log_3 n \rfloor + 1} - 1}{\frac{2}{9} - 1} = \frac{9}{7} n^2 \left(1 - \left(\frac{2}{9}\right)^{\lfloor \log_3 n \rfloor + 1}\right)$$

Kadangi $\left(\frac{2}{9}\right)^{\lfloor \log_3 n \rfloor + 1}$ yra mažėjanti funkcija nuo n tada $1 - \left(\frac{2}{9}\right)^{\lfloor \log_3 n \rfloor + 1} > 1 - \left(\frac{2}{9}\right)^{\lfloor \log_3 3 \rfloor + 1} = \frac{77}{81}$, kai $n > 3$

$$T(n) > \frac{9}{7} n^2 \frac{77}{81} = \frac{11}{9} n^2$$

Parodėme, kad $T(n) = \Omega(n^2)$.

Kadangi galioja $T(n) = O(n^2)$ ir $T(n) = \Omega(n^2)$ galioja ir $T(n) = \Theta(n^2)$, nes $\frac{11}{9}n^2 < T(n) < \frac{9}{7}n^2$ visiems $n > 3$.

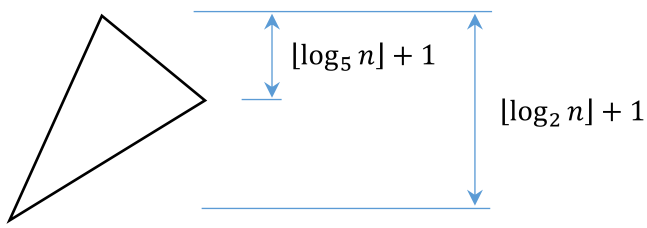
7 uždavinys:

a) Suraskime rekursinės lygties $T(n) = T\left(\frac{n}{2}\right) + T\left(\frac{n}{5}\right) + n^3$ sprendinį.

Žiūrėti 6-10 pav.

Pastebėsime, kad i-tąją medžio eilutę sudaro Niutono binomo nariai ir galime susumuoti, kaip $\left(\frac{1}{2^3} + \frac{1}{5^3}\right)^i = \left(\frac{133}{1000}\right)^i$.

Tolimesni skaičiavimai identiški prieš tai spęsto uždavinio. Medis nėra simetrinis.



Šio uždavinio atveju $\lfloor \log_5 n \rfloor \le h \le \lfloor \log_2 n \rfloor$.

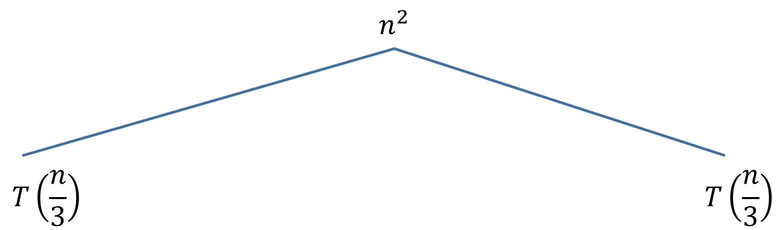
$$T(n) = n^3 \sum_{i=0}^{h} \left(\frac{133}{1000}\right)^i < n^3 \sum_{i=0}^{\infty} \left(\frac{133}{1000}\right)^i = \frac{n^3}{1 - \frac{133}{1000}} = \frac{1000}{867} n^3$$

$$T(n) = n^3 \sum_{i=0}^{h} \left(\frac{133}{1000}\right)^i \ge n^3 \sum_{i=0}^{\lfloor \log_5 n \rfloor} \left(\frac{133}{1000}\right)^i = n^3 \frac{\left(\frac{133}{1000}\right)^{\lfloor \log_5 n \rfloor + 1} - 1}{\frac{133}{1000} - 1} = \frac{1000}{867} n^3 \left(1 - \left(\frac{133}{1000}\right)^{\lfloor \log_5 n \rfloor + 1}\right)$$

$$\ge n^3$$

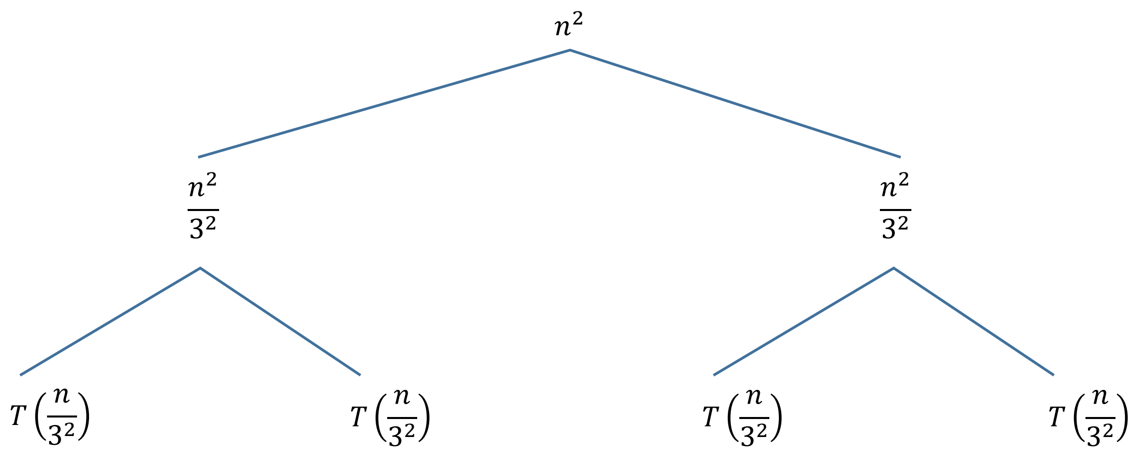
nes $1 - \left(\frac{133}{1000}\right)^{\lfloor \log_5 n \rfloor + 1} \ge \frac{867}{1000}$ visiems $n > 0$.

$T(n) = \Theta(n^3)$, nes $n^3 \le T(n) \le \frac{1000}{867} n^3$ visiems $n > 0$.

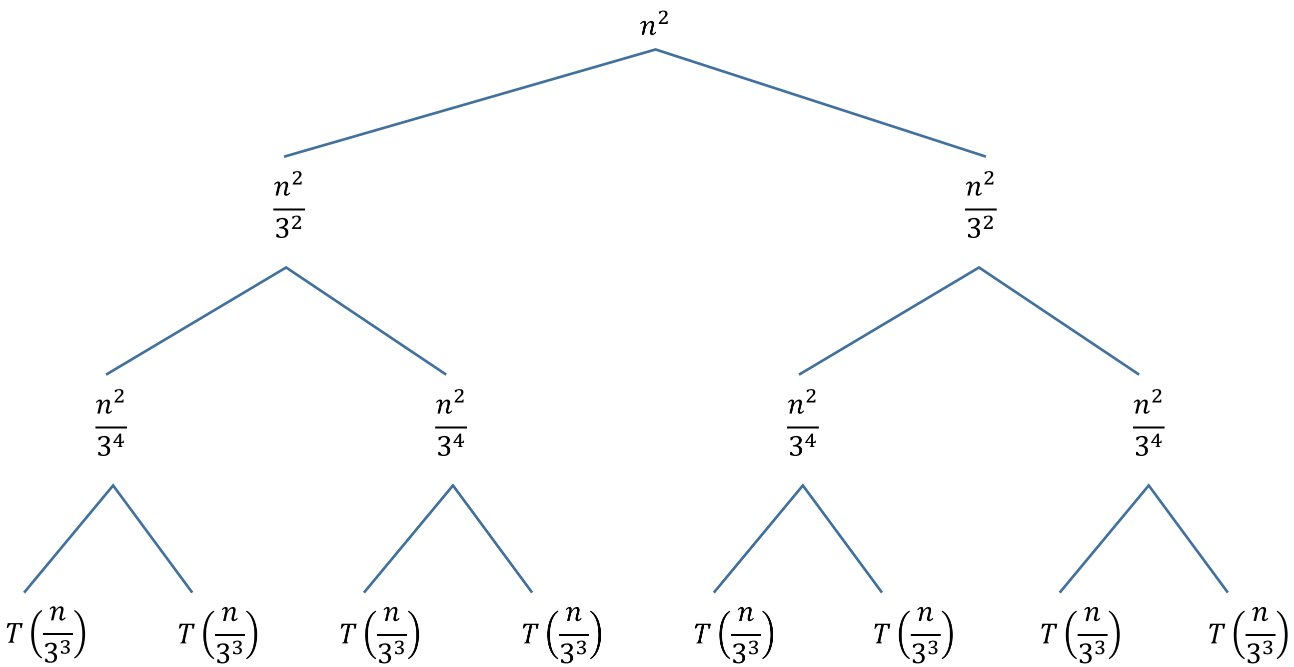

n^2
$T\left(\frac{n}{3}\right)$
$T\left(\frac{n}{3}\right)$

`

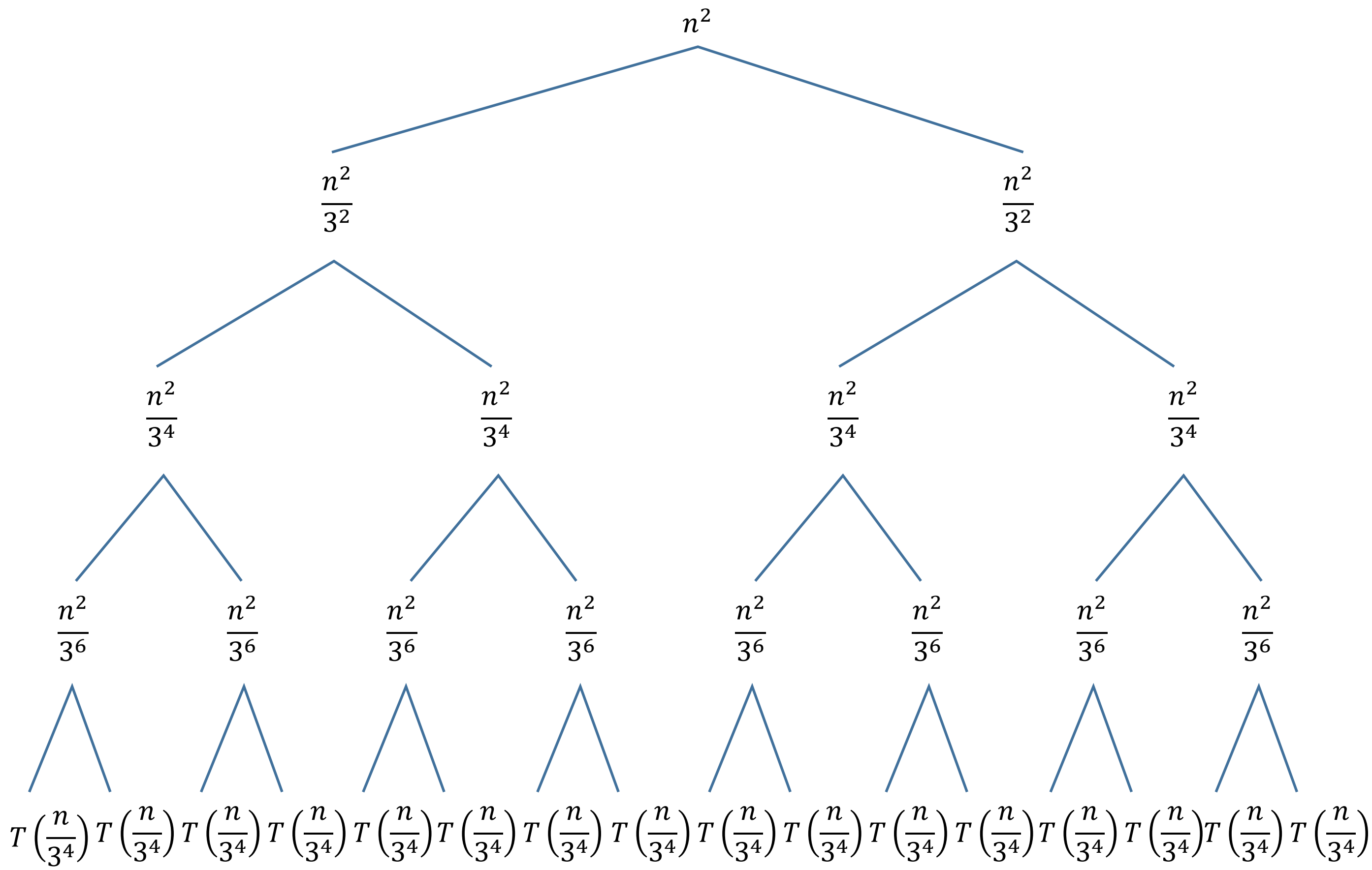
1 pav.


n^2
$\frac{n^2}{3^2}$
$\frac{n^2}{3^2}$
$T\left(\frac{n}{3^2}\right)$
$T\left(\frac{n}{3^2}\right)$
$T\left(\frac{n}{3^2}\right)$
$T\left(\frac{n}{3^2}\right)$

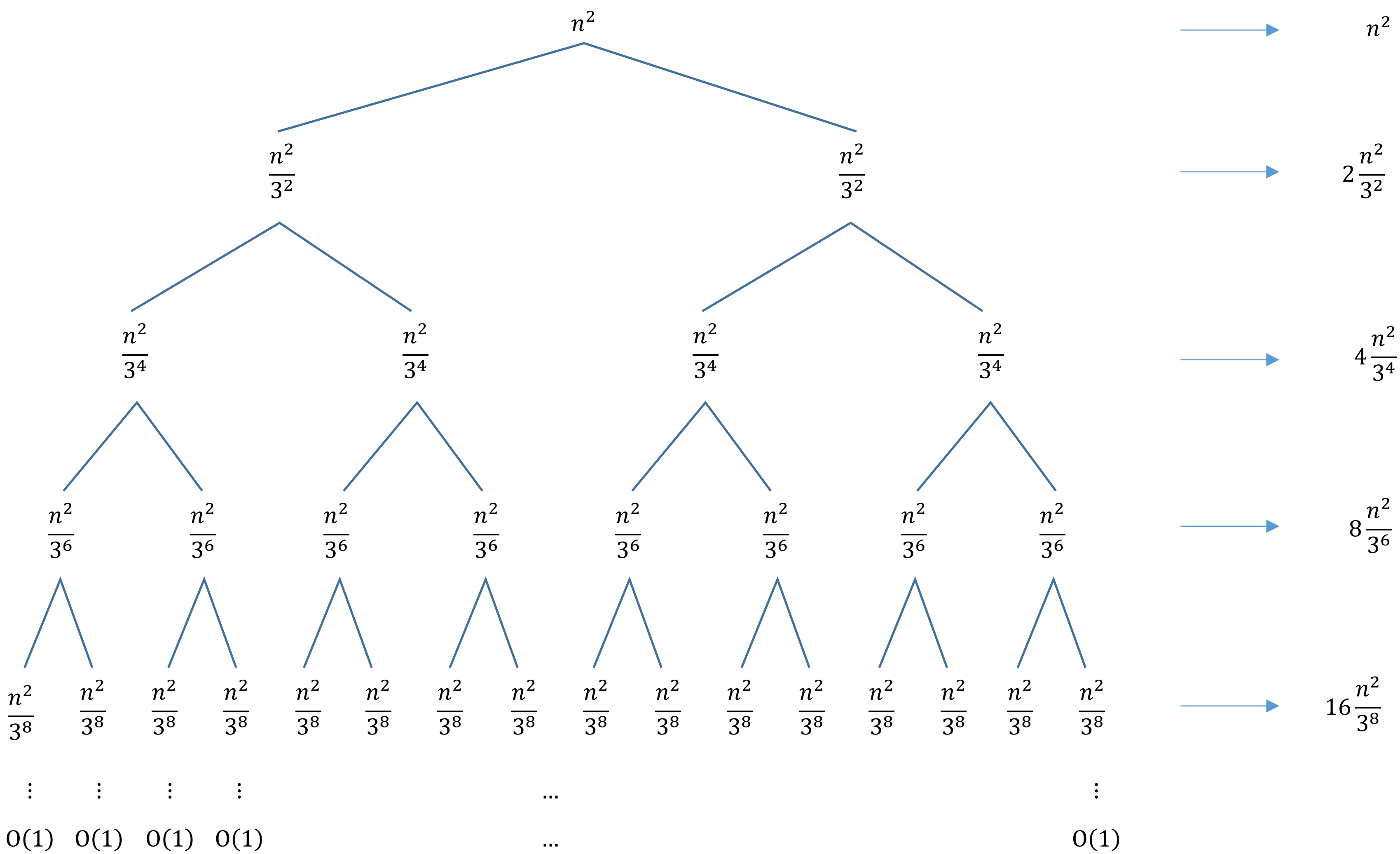
2 pav.


n^2
$\frac{n^2}{3^2}$
$\frac{n^2}{3^2}$
$\frac{n^2}{3^4}$
$\frac{n^2}{3^4}$
$\frac{n^2}{3^4}$
$\frac{n^2}{3^4}$
$T\left(\frac{n}{3^3}\right)$
$T\left(\frac{n}{3^3}\right)$
$T\left(\frac{n}{3^3}\right)$
$T\left(\frac{n}{3^3}\right)$
$T\left(\frac{n}{3^3}\right)$
$T\left(\frac{n}{3^3}\right)$
$T\left(\frac{n}{3^3}\right)$
$T\left(\frac{n}{3^3}\right)$

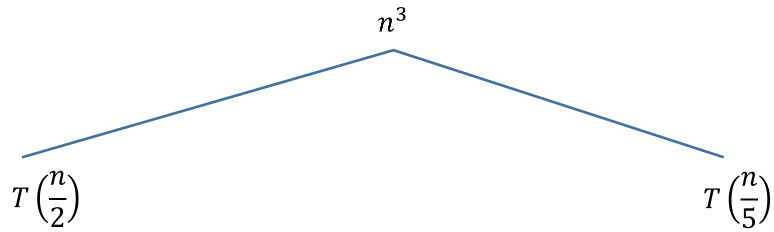
3 pav.


n^2
$\frac{n^2}{3^2}$ $\frac{n^2}{3^2}$
$\frac{n^2}{3^4}$ $\frac{n^2}{3^4}$ $\frac{n^2}{3^4}$ $\frac{n^2}{3^4}$
$\frac{n^2}{3^6}$ $\frac{n^2}{3^6}$ $\frac{n^2}{3^6}$ $\frac{n^2}{3^6}$ $\frac{n^2}{3^6}$ $\frac{n^2}{3^6}$ $\frac{n^2}{3^6}$ $\frac{n^2}{3^6}$
$T\left(\frac{n}{3^4}\right) T\left(\frac{n}{3^4}\right) T\left(\frac{n}{3^4}\right) T\left(\frac{n}{3^4}\right) T\left(\frac{n}{3^4}\right) T\left(\frac{n}{3^4}\right) T\left(\frac{n}{3^4}\right) T\left(\frac{n}{3^4}\right) T\left(\frac{n}{3^4}\right) T\left(\frac{n}{3^4}\right) T\left(\frac{n}{3^4}\right) T\left(\frac{n}{3^4}\right) T\left(\frac{n}{3^4}\right) T\left(\frac{n}{3^4}\right) T\left(\frac{n}{3^4}\right) T\left(\frac{n}{3^4}\right)$

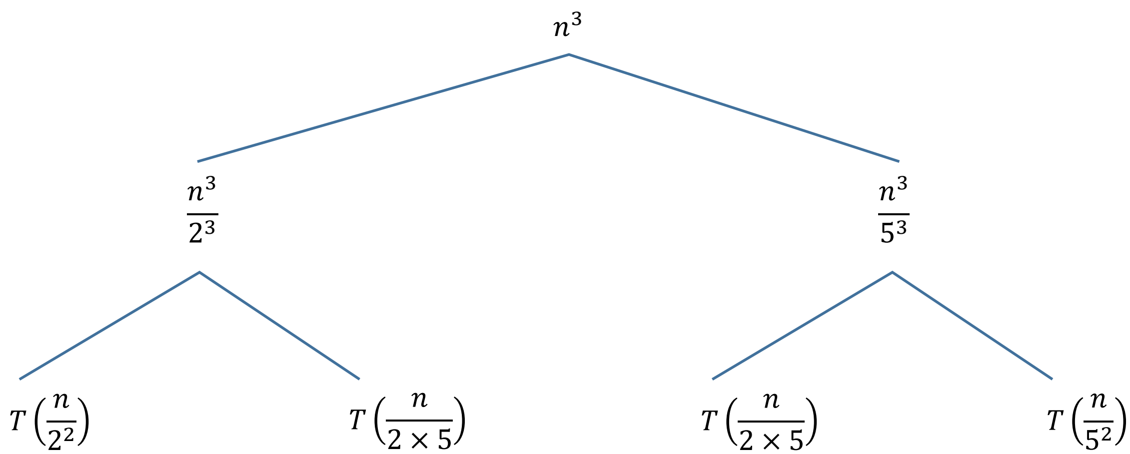
4 pav.



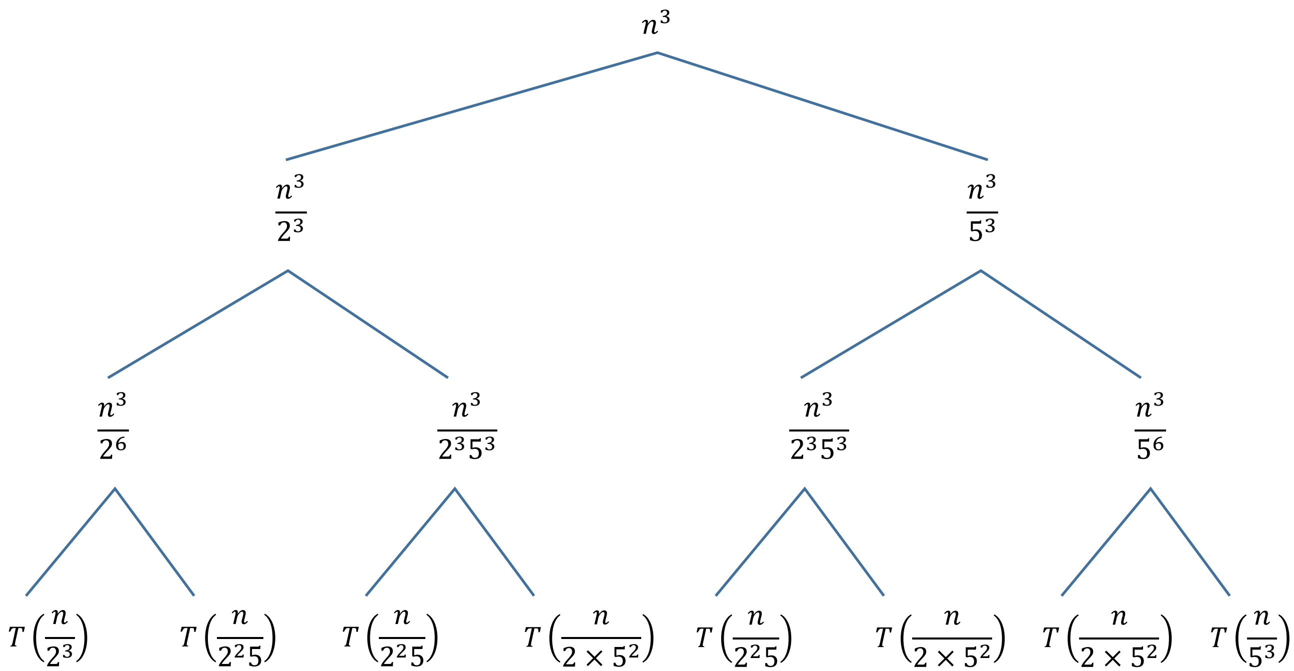
5 pav.


n^3
$T\left(\frac{n}{2}\right)$
$T\left(\frac{n}{5}\right)$

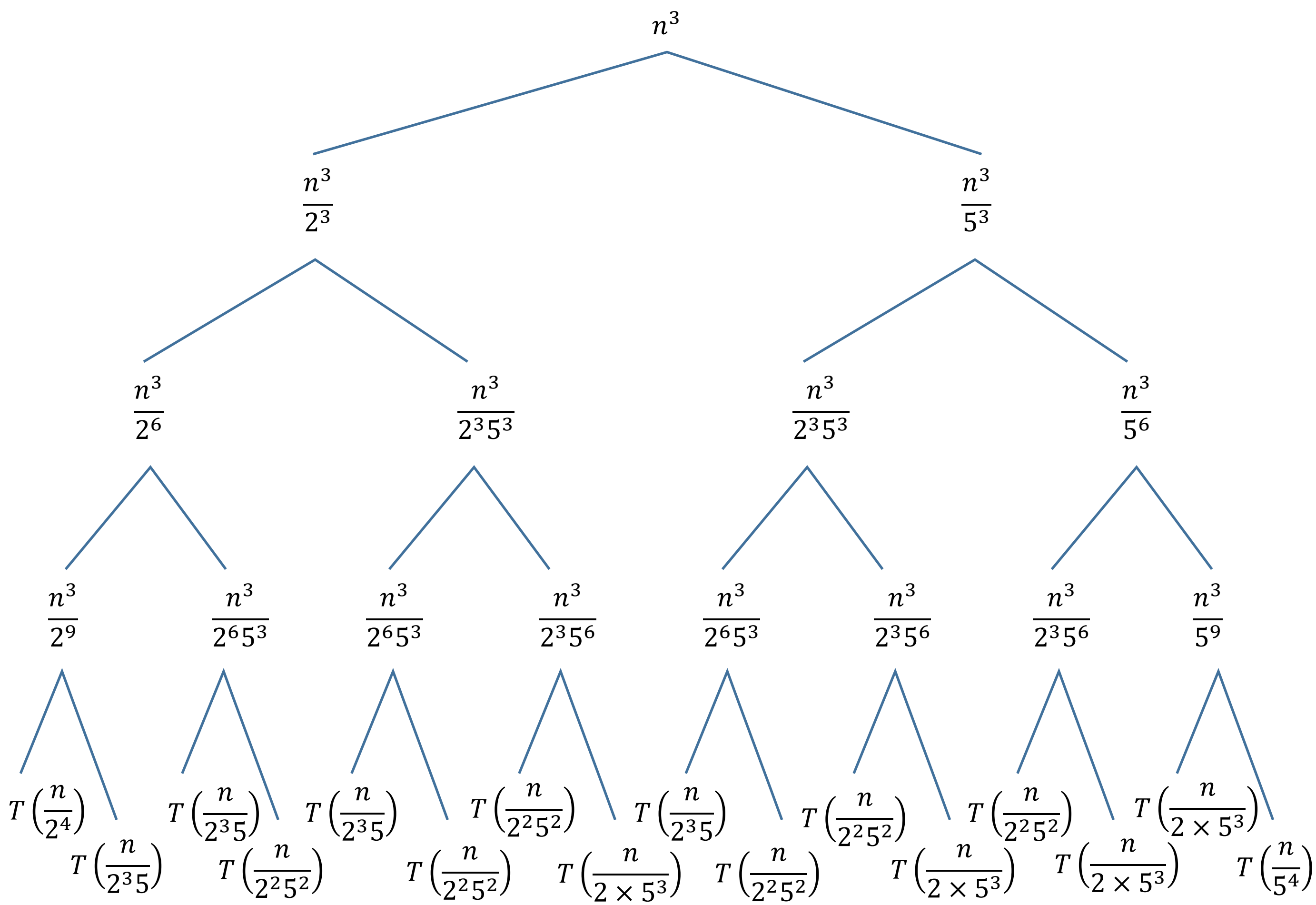
6 pav.


n^3
$\frac{n^3}{2^3}$
$\frac{n^3}{5^3}$
$T\left(\frac{n}{2^2}\right)$
$T\left(\frac{n}{2 \times 5}\right)$
$T\left(\frac{n}{2 \times 5}\right)$
$T\left(\frac{n}{5^2}\right)$

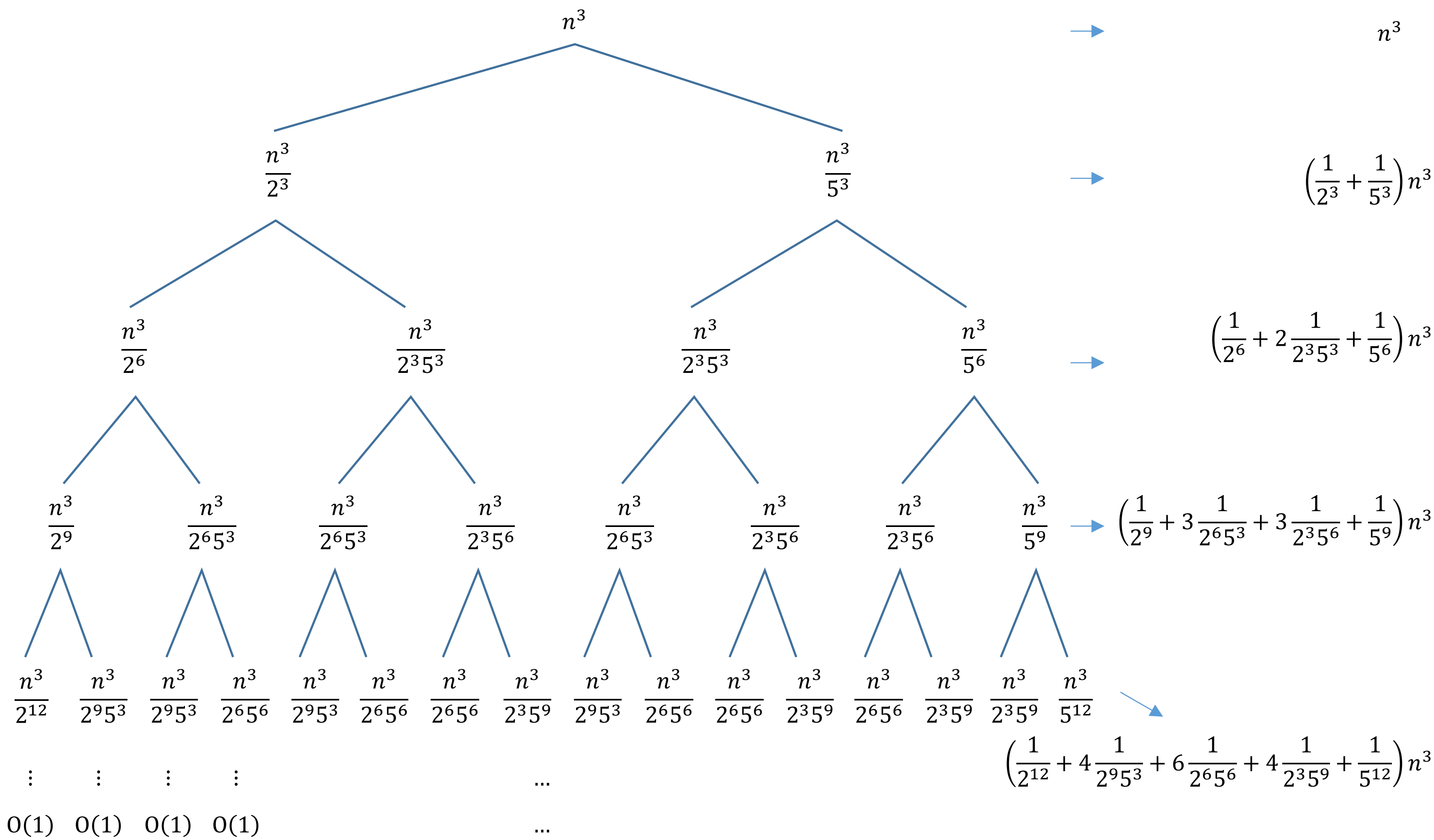
7 pav.


n^3
$\frac{n^3}{2^3}$
$\frac{n^3}{5^3}$
$\frac{n^3}{2^6}$
$\frac{n^3}{2^3 5^3}$
$\frac{n^3}{2^3 5^3}$
$\frac{n^3}{5^6}$
$T\left(\frac{n}{2^3}\right)$
$T\left(\frac{n}{2^2 5}\right)$
$T\left(\frac{n}{2^2 5}\right)$
$T\left(\frac{n}{2 \times 5^2}\right)$
$T\left(\frac{n}{2^2 5}\right)$
$T\left(\frac{n}{2 \times 5^2}\right)$
$T\left(\frac{n}{2 \times 5^2}\right)$
$T\left(\frac{n}{5^3}\right)$

8 pav.


n^3
$\frac{n^3}{2^3}$
$\frac{n^3}{5^3}$
$\frac{n^3}{2^6}$
$\frac{n^3}{2^3 5^3}$
$\frac{n^3}{2^3 5^3}$
$\frac{n^3}{5^6}$
$\frac{n^3}{2^9}$
$\frac{n^3}{2^6 5^3}$
$\frac{n^3}{2^6 5^3}$
$\frac{n^3}{2^3 5^6}$
$\frac{n^3}{2^6 5^3}$
$\frac{n^3}{2^3 5^6}$
$\frac{n^3}{2^3 5^6}$
$\frac{n^3}{5^9}$
$T\left(\frac{n}{2^4}\right)$
$T\left(\frac{n}{2^3 5}\right)$
$T\left(\frac{n}{2^3 5}\right)$
$T\left(\frac{n}{2^2 5^2}\right)$
$T\left(\frac{n}{2^3 5}\right)$
$T\left(\frac{n}{2^2 5^2}\right)$
$T\left(\frac{n}{2^2 5^2}\right)$
$T\left(\frac{n}{2 \times 5^3}\right)$
$T\left(\frac{n}{2^3 5}\right)$
$T\left(\frac{n}{2^2 5^2}\right)$
$T\left(\frac{n}{2^2 5^2}\right)$
$T\left(\frac{n}{2 \times 5^3}\right)$
$T\left(\frac{n}{2^2 5^2}\right)$
$T\left(\frac{n}{2 \times 5^3}\right)$
$T\left(\frac{n}{2 \times 5^3}\right)$
$T\left(\frac{n}{5^4}\right)$

9 pav.


n^3
$\frac{n^3}{2^3}$ $\frac{n^3}{5^3}$
$\frac{n^3}{2^6}$ $\frac{n^3}{2^3 5^3}$ $\frac{n^3}{2^3 5^3}$ $\frac{n^3}{5^6}$
$\frac{n^3}{2^9}$ $\frac{n^3}{2^6 5^3}$ $\frac{n^3}{2^6 5^3}$ $\frac{n^3}{2^3 5^6}$ $\frac{n^3}{2^6 5^3}$ $\frac{n^3}{2^3 5^6}$ $\frac{n^3}{2^3 5^6}$ $\frac{n^3}{5^9}$
$\frac{n^3}{2^{12}}$ $\frac{n^3}{2^9 5^3}$ $\frac{n^3}{2^9 5^3}$ $\frac{n^3}{2^6 5^6}$ $\frac{n^3}{2^9 5^3}$ $\frac{n^3}{2^6 5^6}$ $\frac{n^3}{2^6 5^6}$ $\frac{n^3}{2^3 5^9}$ $\frac{n^3}{2^9 5^3}$ $\frac{n^3}{2^6 5^6}$ $\frac{n^3}{2^6 5^6}$ $\frac{n^3}{2^3 5^9}$ $\frac{n^3}{2^6 5^6}$ $\frac{n^3}{2^3 5^9}$ $\frac{n^3}{2^3 5^9}$ $\frac{n^3}{5^{12}}$
⋮ ⋮ ⋮ ⋮ ...
O(1) O(1) O(1) O(1) ...
n^3
$\left(\frac{1}{2^3} + \frac{1}{5^3}\right) n^3$
$\left(\frac{1}{2^6} + 2\frac{1}{2^3 5^3} + \frac{1}{5^6}\right) n^3$
$\left(\frac{1}{2^9} + 3\frac{1}{2^6 5^3} + 3\frac{1}{2^3 5^6} + \frac{1}{5^9}\right) n^3$
$\left(\frac{1}{2^{12}} + 4\frac{1}{2^9 5^3} + 6\frac{1}{2^6 5^6} + 4\frac{1}{2^3 5^9} + \frac{1}{5^{12}}\right) n^3$

10 pav.

8 uždavinys: Įvertinti žemiau pateiktos procedūros sudėtingumą:

```
class MyDataArray
{
    protected int length;
    public int Length
        {
           get
           {
               return length;
           }
        }

    double[] data;
    public MyDataArray(int n)
    {
        data = new double[n];
        length = n;
        Random rand = new Random();
        for (int i = 0; i < length; i++)
        {
            data[i] = rand.NextDouble();
        }
    }

    public double this[int index]
    {
        get { return data[index]; }
    }

    public void Swap(int j, double a, double b)
    {
        data[j - 1] = a;
        data[j] = b;
    }
}

static void BubbleSort(MyDataArray items)
{
     double prevdata, currentdata;
     for (int i = items.Length - 1; i >= 0; i--)

        {
         currentdata = items[0];
         for (int j = 1; j <= i; j++)
         {
             prevdata = currentdata;
             currentdata = items[j];
             if (prevdata > currentdata)
             {
                 items.Swap(j, currentdata, prevdata);
                     currentdata = prevdata;
            }
        }
    }
}
```

Atliekant įvertinmus su duomenų struktūromos ir objektai reikia atkreipti dėmesį, kad tiek klasių metodus tiek savybes reikia laikyti kaip procedūras ir jas įvertinti iš anksto. *BubbleSort* naudoja `MyDataArray` klasės `Length, indeksavimo operatorių []` ir `Swap.`

```
    class MyDataArray                                                    Kaina | Kartai
    {                                                                          |
        protected int length;                                                  |
        public int Length                                                      |
            {                                                                  |
               get                                                             |
               {                                                               |
                   return length;                                         c1   |  1
               }                                                               |
            }                                                                  |
                                                                               |
        double[] data;                                                         |
        public MyDataArray(int n)                                              |
        {                                                                      |
            data = new double[n];                                              |
            length = n;                                                        |
            Random rand = new Random();                                        |
            for (int i = 0; i < length; i++)                                   |
            {                                                                  |
                data[i] = rand.NextDouble();                                   |
            }                                                                  |
        }                                                                      |
                                                                               |
        public double this[int index]                                          |
        {                                                                      |
            get { return data[index]; }                                   c2   |  1
        }                                                                      |
                                                                               |
        public void Swap(int j, double a, double b)                            |
        {                                                                      |
            data[j - 1] = a;                                              c3   |  1
            data[j] = b;                                                  c3   |  1
        }                                                                      |
    }                                                                          |
}
```

MyDataArray klasės atributo/savybės *Length* get įvertinimas $T_L(obj\_\text{MyDataArray}) = c_1$, atributo pagal nutylėjimą $T_I(obj\_\text{MyDataArray}, j) = c_2$, o metodo *Swap* sudėtingumas $T_S(obj\_\text{MyDataArray}, j) = 2c_3$.

`static void BubbleSort(MyDataArray items)`		
`{`		
`    double prevdata, currentdata;`	c_4	1
`    for (int i = items.Length - 1; i >= 0; i--)`	$T_L(items)$	1
	c_5	$\text{items.Length} + 1$
`    {`		
`        currentdata = items[0];`	$c_4 + T_I(items, 0)$	items.Length
`        for (int j = 1; j <= i; j++)`	c_5	$\sum_{i=0}^{items.Length-1} \sum_{j=1}^{i+1} 1$
`        {`		
`            prevdata = currentdata;`	c_7	$\sum_{i=0}^{items.Length-1} \sum_{j=1}^{i} 1$
`            currentdata = items[j];`	$c_4 + T_I(items, j)$	$\sum_{i=0}^{items.Length-1} \sum_{j=1}^{i} 1$
`            if (prevdata > currentdata)`	c_6	$\sum_{i=0}^{items.Length-1} \sum_{j=1}^{i} 1$
`            {`		
`                items.Swap(j, currentdata, prevdata);`	$c_4 + T_S(items, j)$	$\sum_{i=0}^{items.Length-1} \sum_{j=1}^{i} t_j$
`                currentdata = prevdata;`	c_7	$\sum_{i=0}^{items.Length-1} \sum_{j=1}^{i} t_j$
`            }`		
`        }`		
`    }`		
`}`		

Raskime tarpines sumas:

$$\sum_{i=0}^{items.Length-1} \sum_{j=1}^{i+1} 1 = \sum_{i=0}^{items.Length-1} (i+1) = \frac{items.Length(items.Length-1)}{2} + items.Length$$

$$= \frac{items.Length(items.Length+1)}{2}$$

$$\sum_{i=0}^{items.Length-1} \sum_{j=1}^{i} 1 = \sum_{i=0}^{items.Length-1} i = \frac{items.Length(items.Length-1)}{2}$$

$$0 \leq \sum_{i=0}^{items.Length-1} \sum_{j=1}^{i} t_j \leq \sum_{i=0}^{items.Length-1} \sum_{j=1}^{i} 1 = \frac{items.Length(items.Length-1)}{2}$$

čia $t_j = 1$, jei sąlyga $prevdata > currentdata$ tenkinama, o kitu atveju $t_j = 0$.

Tokiu atveju

$$
\begin{aligned}
&T_{Bubble}(obj\_\mathrm{MyDataArray}) \\
&= c_4 + T_L(obj\_\mathrm{MyDataArray}) + c_5(items.Length + 1) + \left(c_4 + T_I(items, 0)\right)items.Length \\
&+ c_5 \frac{items.Length(items.Length + 1)}{2} + c_7 \frac{items.Length(items.Length - 1)}{2} \\
&+ \left(c_4 + T_I(items, j)\right) \frac{items.Length(items.Length - 1)}{2} \\
&+ c_6 \frac{items.Length(items.Length - 1)}{2} + \left(c_4 + T_S(items, j)\right) \sum_{i=0}^{items.Length-1} \sum_{j=1}^{i} t_j \\
&+ c_7 \sum_{i=0}^{items.Length-1} \sum_{j=1}^{i} t_j \\
&= c_4 + c_1 + c_5(items.Length + 1) + (c_4 + c_2)items.Length \\
&+ c_5 \frac{items.Length(items.Length + 1)}{2} + c_7 \frac{items.Length(items.Length - 1)}{2} \\
&+ (c_4 + c_2) \frac{items.Length(items.Length - 1)}{2} + c_6 \frac{items.Length(items.Length - 1)}{2} \\
&+ (c_4 + 2c_3) \sum_{i=0}^{items.Length-1} \sum_{j=1}^{i} t_j + c_7 \sum_{i=0}^{items.Length-1} \sum_{j=1}^{i} t_j \\
&= \frac{c_2 + c_4 + c_5 + c_6 + c_7}{2} items.Length^2 \\
&+ \left(c_2 + c_4 + c_5 + \frac{c_5 - c_2 - c_4 - c_6 - c_7}{2}\right) items.Length \\
&+ (c_4 + 2c_3 + c_7) \sum_{i=0}^{items.Length-1} \sum_{j=1}^{i} t_j + c_1 + c_4 + c_5 = \Theta(items.Length^2)
\end{aligned}
$$

nes nekeičia svariausio dėmes eilės sumos $\sum_{i=0}^{items.Length-1} \sum_{j=1}^{i} t_j$ įvertinimas tiek iš viršaus tiek iš apačios. Kadangi `MyDataArray` klasės konstruktorius turi parametrą n, kuris nusako generuojamų duomenų kiekį, t. y. $Length = n$, tai $T_{Bubble}(obj\_\mathrm{MyDataArray}) = \Theta(n^2)$.

Išvada, rikiavimas Burbuliuko metodu geriausiu atveju veikia blogiau nei rikiavimas įterpiant.