Enumeration Sort

Peterson Wagner Kava de Carvalho

Enumeration Sort

Ordenação por ranks

input	10	5	20	2	8
rank	3	1	4	0	2
result	2	5	8	10	20

Enumeration Sort

10	5	20	2	8

10

5

20

2

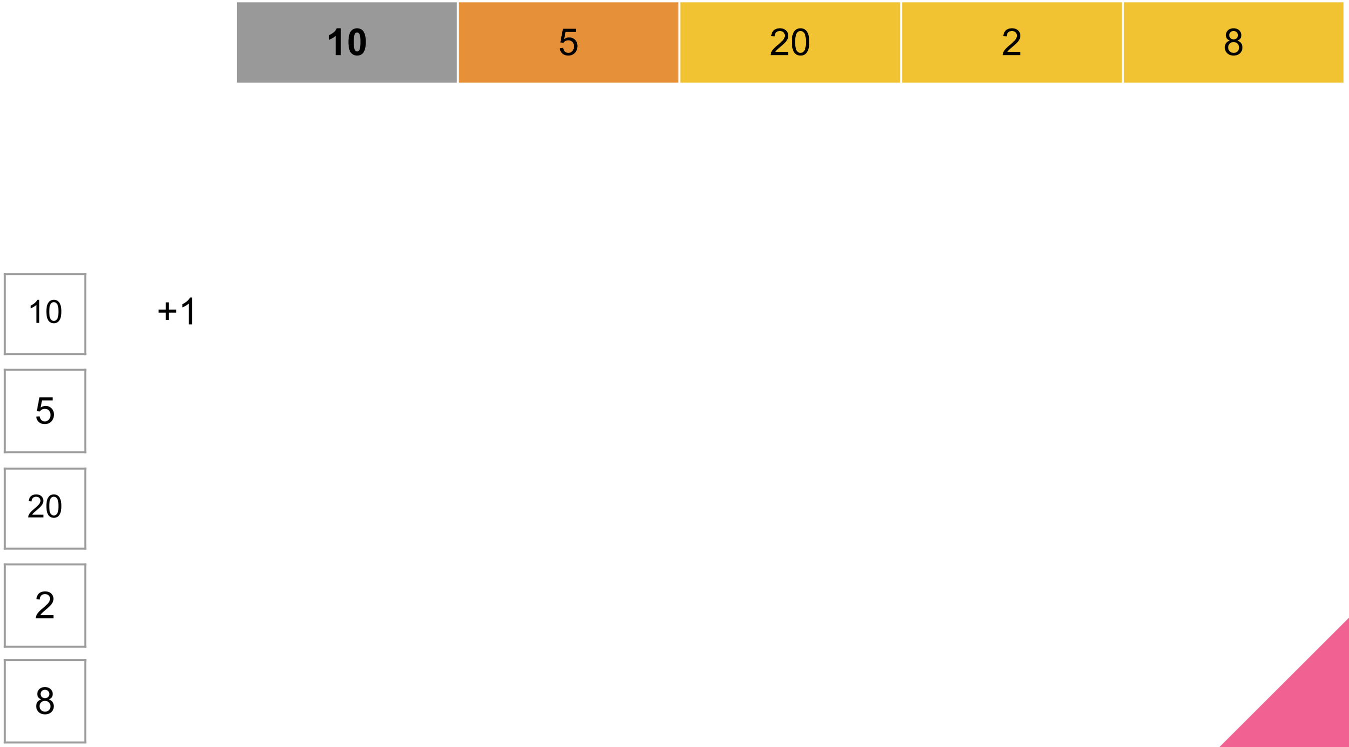
8

Enumeration Sort

10	5	20	2	8

10
5
20
2
8

Enumeration Sort



10	5	20	2	8

10 +1

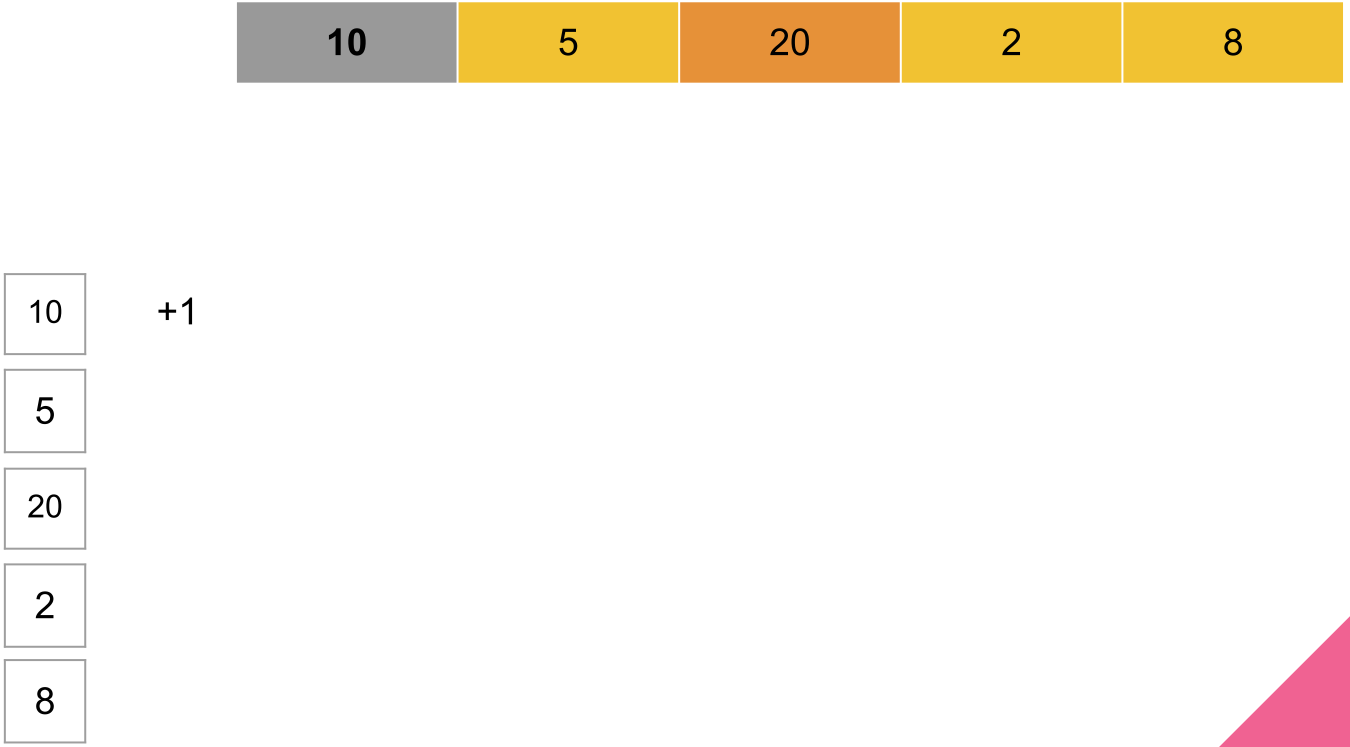
5

20

2

8

Enumeration Sort



10	5	20	2	8

10 +1

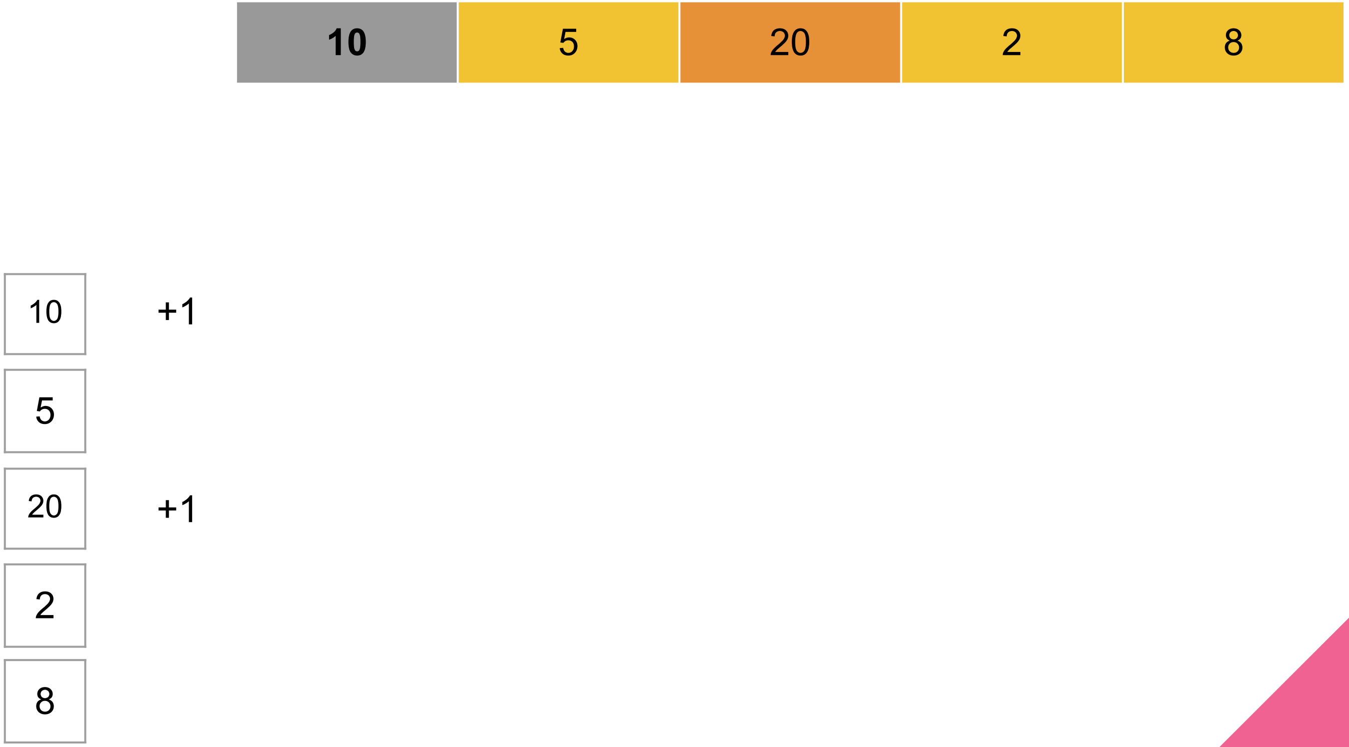
5

20

2

8

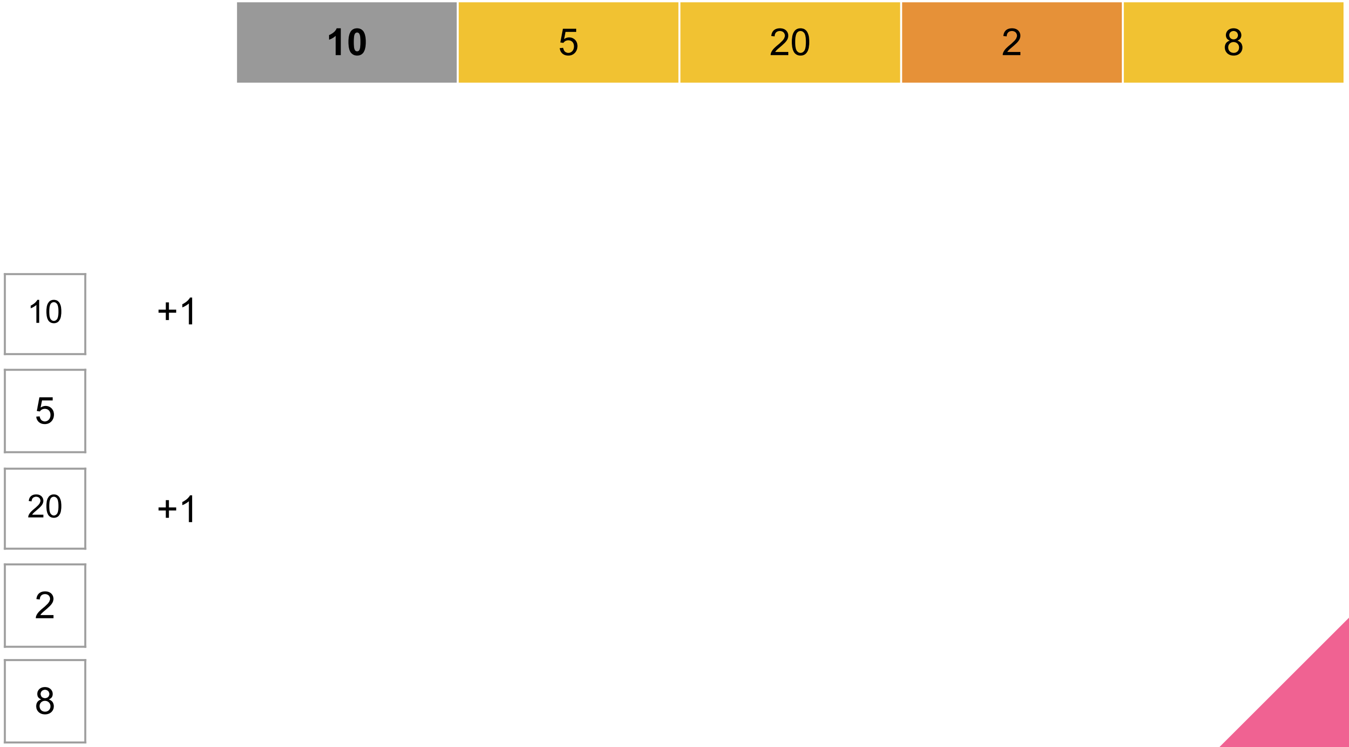
Enumeration Sort



10	5	20	2	8

10	+1
5	
20	+1
2	
8	

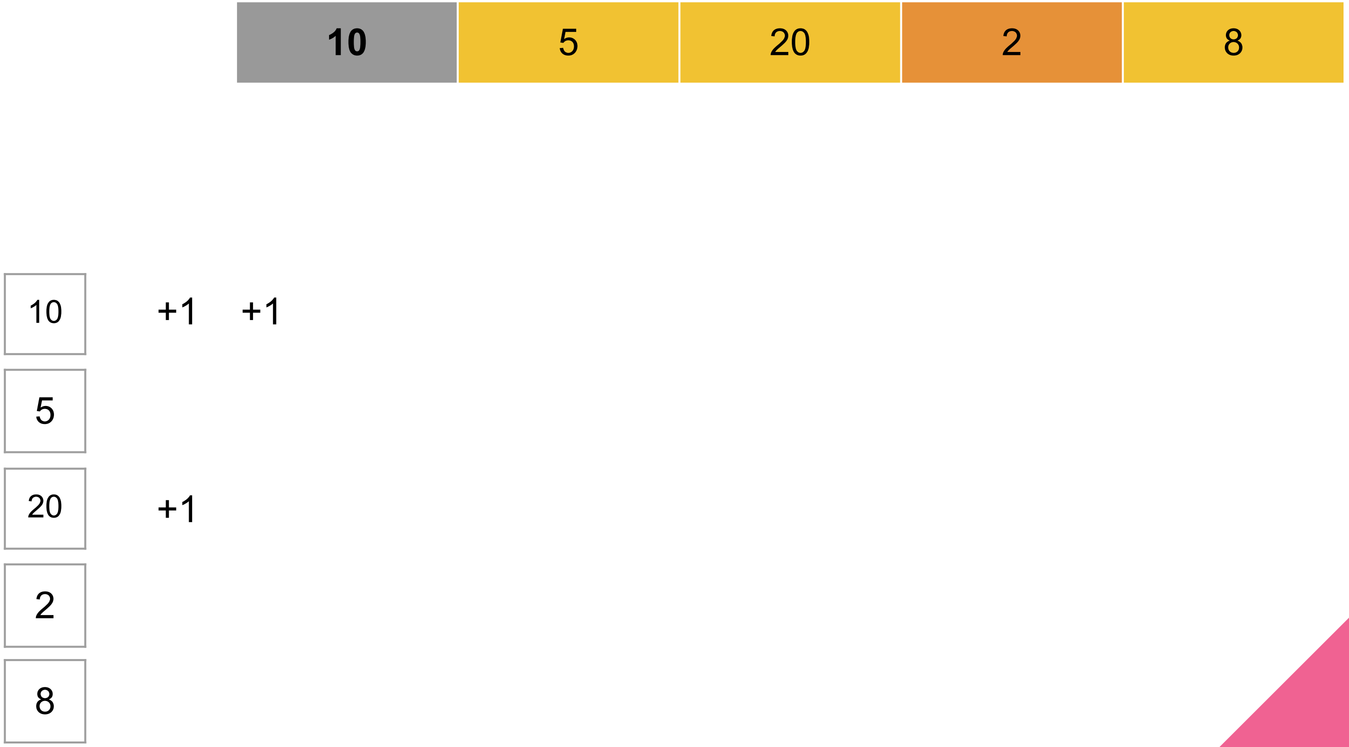
Enumeration Sort



10	5	20	2	8

10	+1
5	
20	+1
2	
8	

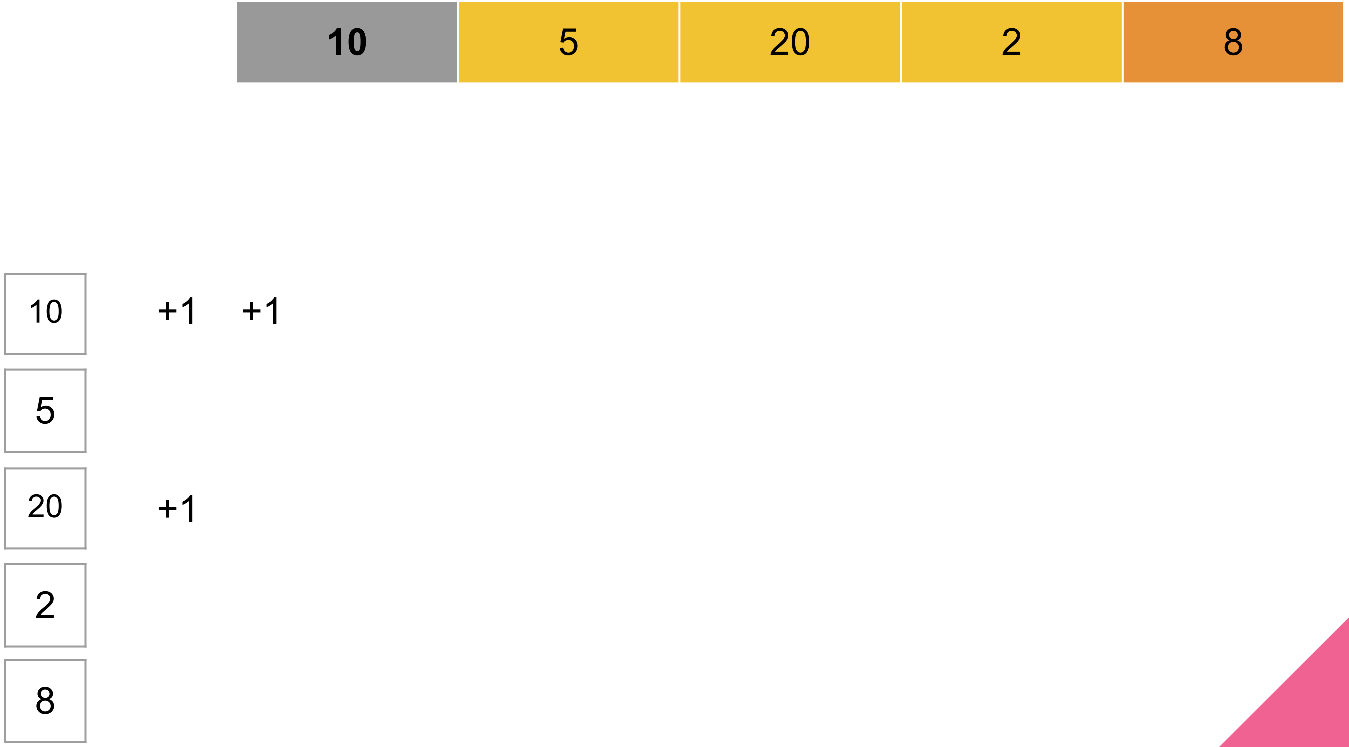
Enumeration Sort



10	5	20	2	8

10	+1	+1
5		
20	+1	
2		
8		

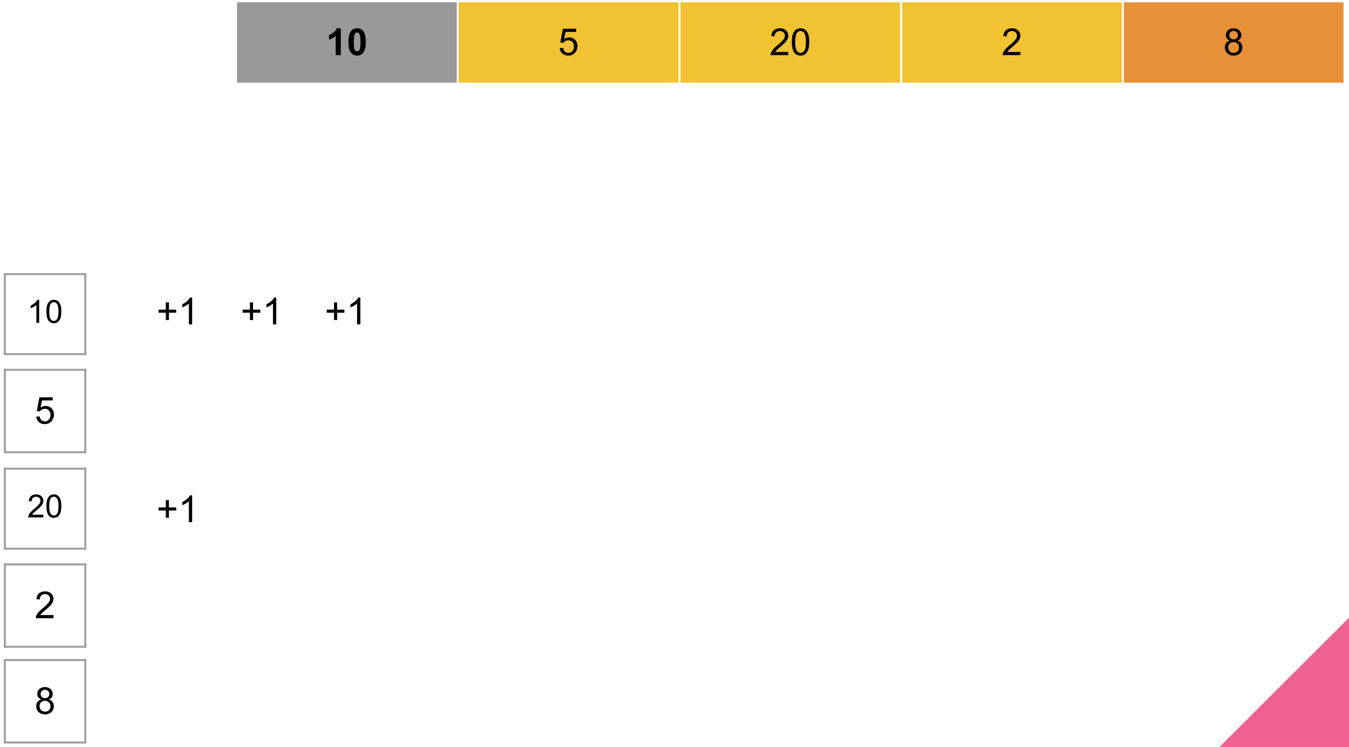
Enumeration Sort



10	5	20	2	8

10	+1	+1
5		
20	+1	
2		
8		

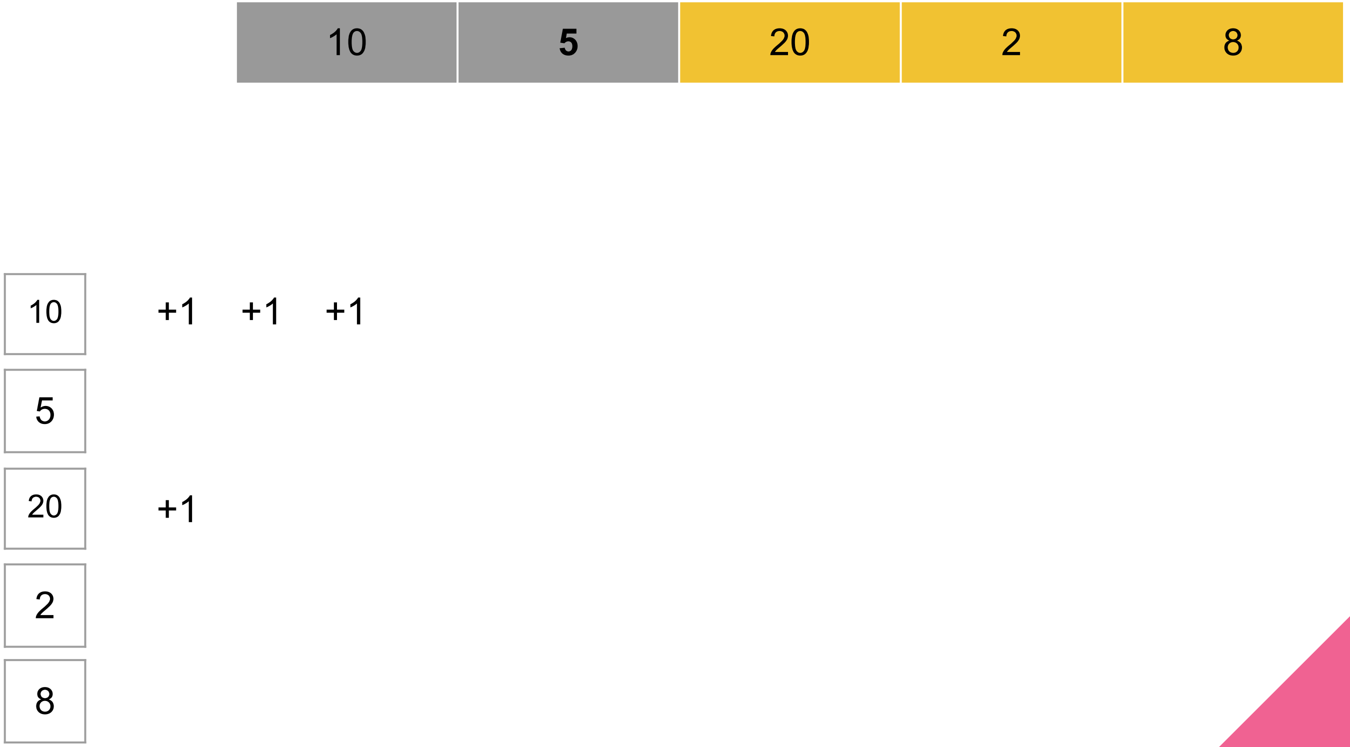
Enumeration Sort



10	5	20	2	8

10	+1	+1	+1
5			
20	+1		
2			
8			

Enumeration Sort



10	**5**	20	2	8

10 +1 +1 +1

5

20 +1

2

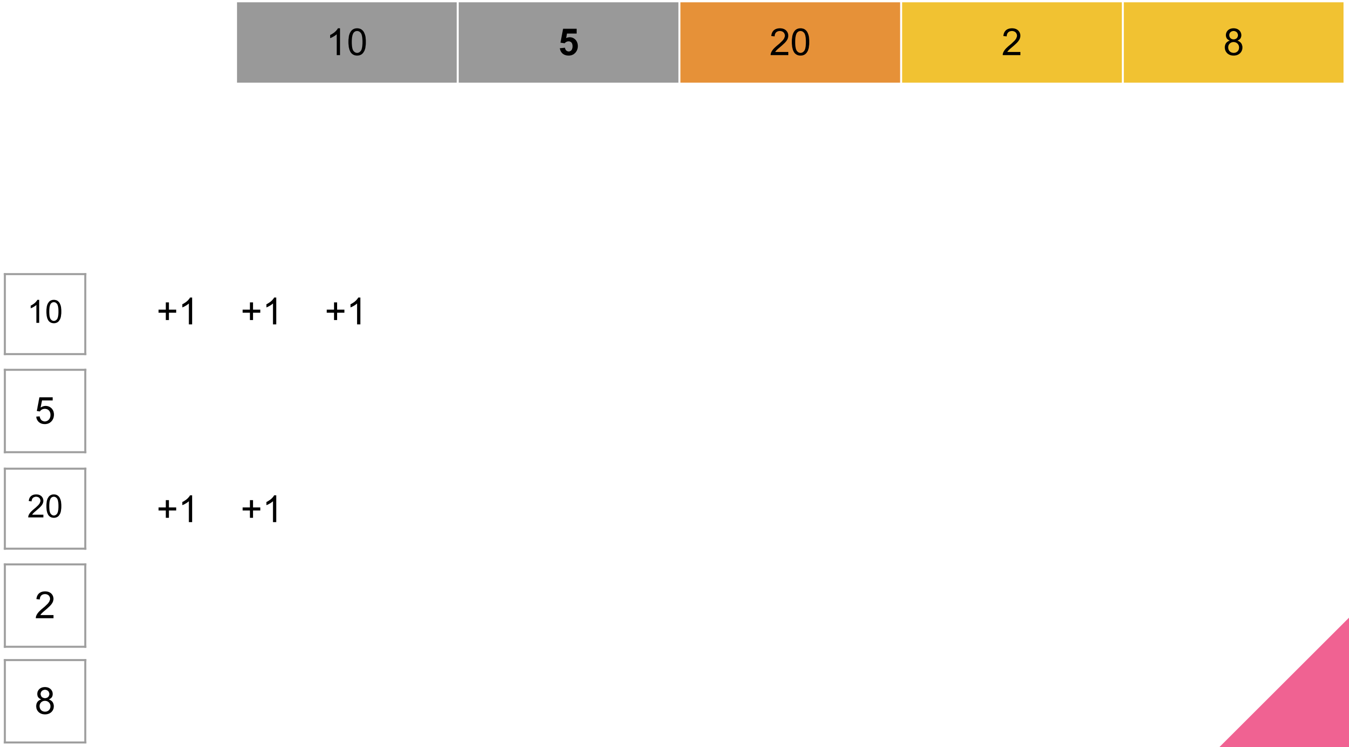
8

Enumeration Sort

10	**5**	20	2	8

10	+1	+1	+1
5			
20	+1		
2			
8			

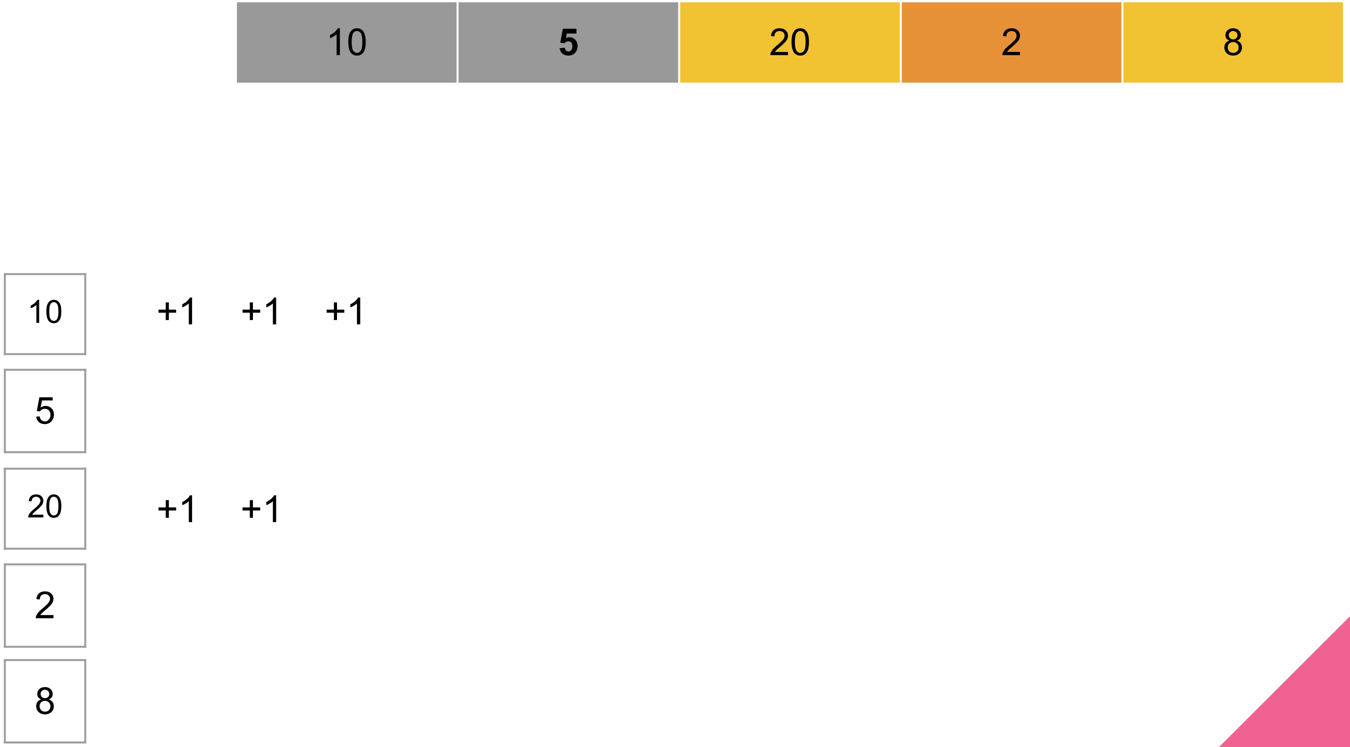
Enumeration Sort



10	**5**	20	2	8

10	+1	+1	+1
5			
20	+1	+1	
2			
8			

Enumeration Sort



10	**5**	20	2	8

10	+1	+1	+1
5			
20	+1	+1	
2			
8			

Enumeration Sort

10	**5**	20	2	8

10	+1	+1	+1
5	+1		
20	+1	+1	
2			
8			

Enumeration Sort

10	**5**	20	2	8

10	+1	+1	+1
5	+1		
20	+1	+1	
2			
8			

Enumeration Sort

10	**5**	20	2	8

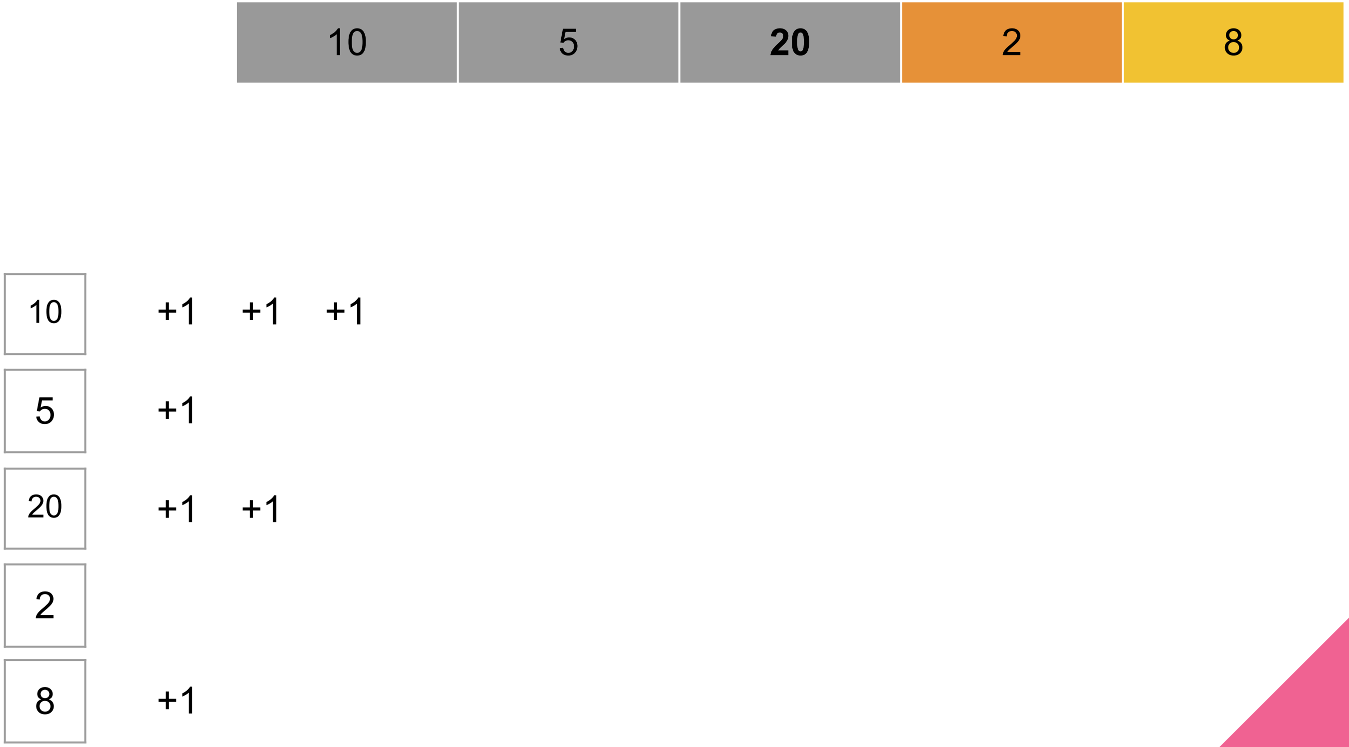
10	+1	+1	+1
5	+1		
20	+1	+1	
2			
8	+1		

Enumeration Sort

10	5	**20**	2	8

10	+1	+1	+1
5	+1		
20	+1	+1	
2			
8	+1		

Enumeration Sort



10	5	**20**	2	8

10	+1	+1	+1
5	+1		
20	+1	+1	
2			
8	+1		

Enumeration Sort

10	5	**20**	2	8

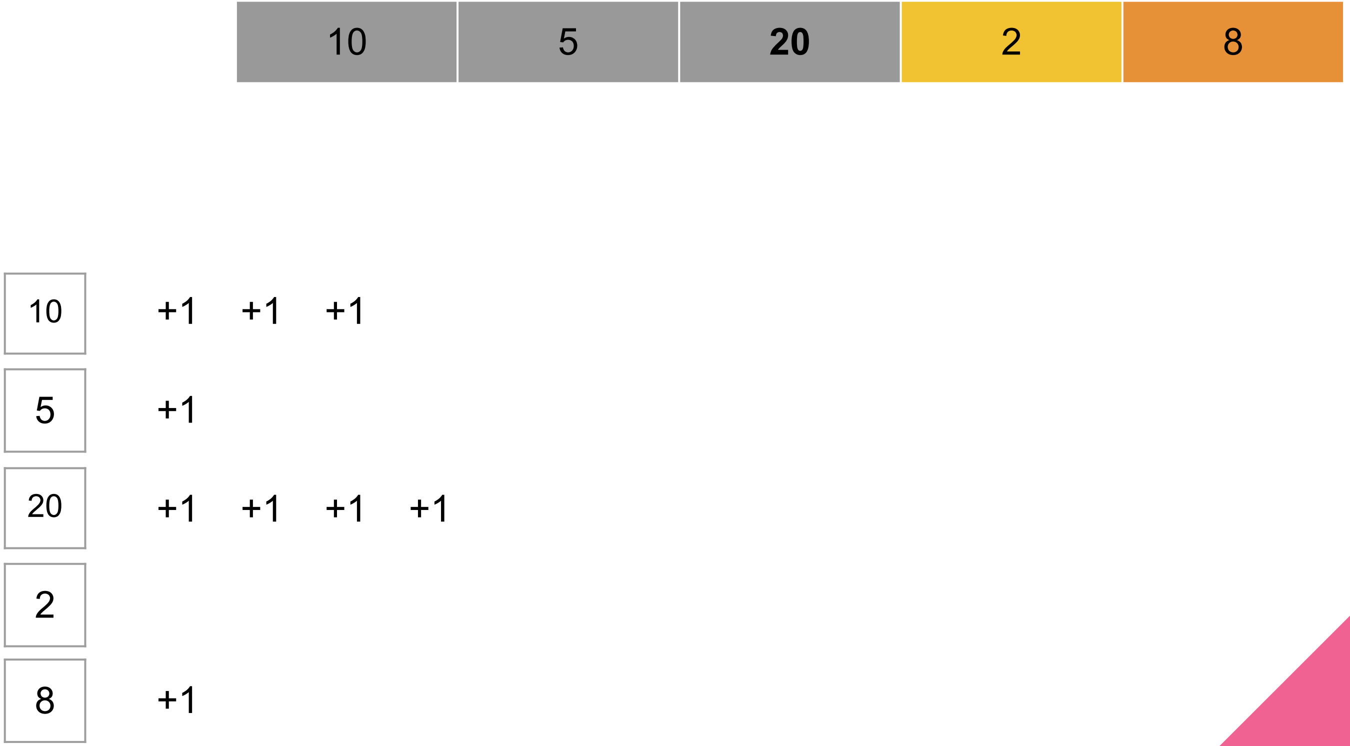
10	+1	+1	+1
5	+1		
20	+1	+1	+1
2			
8	+1		

Enumeration Sort

10	5	**20**	2	8

10	+1	+1	+1
5	+1		
20	+1	+1	+1
2			
8	+1		

Enumeration Sort



10	5	**20**	2	8

10	+1	+1	+1	
5	+1			
20	+1	+1	+1	+1
2				
8	+1			

Enumeration Sort

10	5	20	**2**	8

10	+1	+1	+1	
5	+1			
20	+1	+1	+1	+1
2				
8	+1			

Enumeration Sort

10	5	20	**2**	8

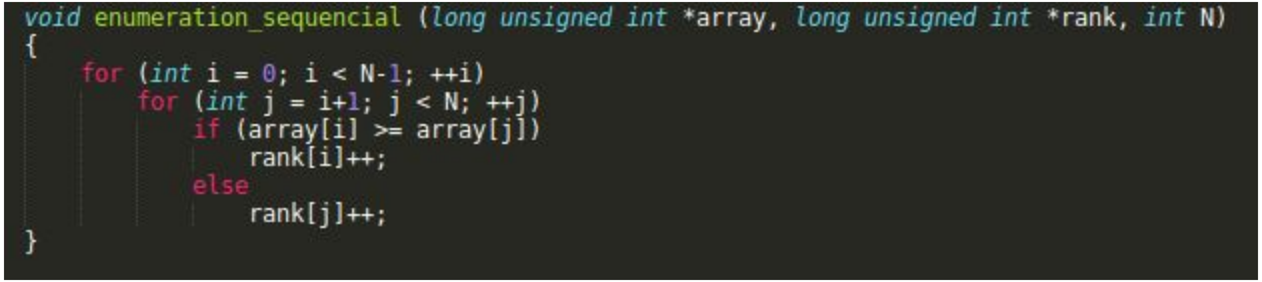
10	+1	+1	+1	
5	+1			
20	+1	+1	+1	+1
2				
8	+1			

Enumeration Sort

10	5	20	2	8

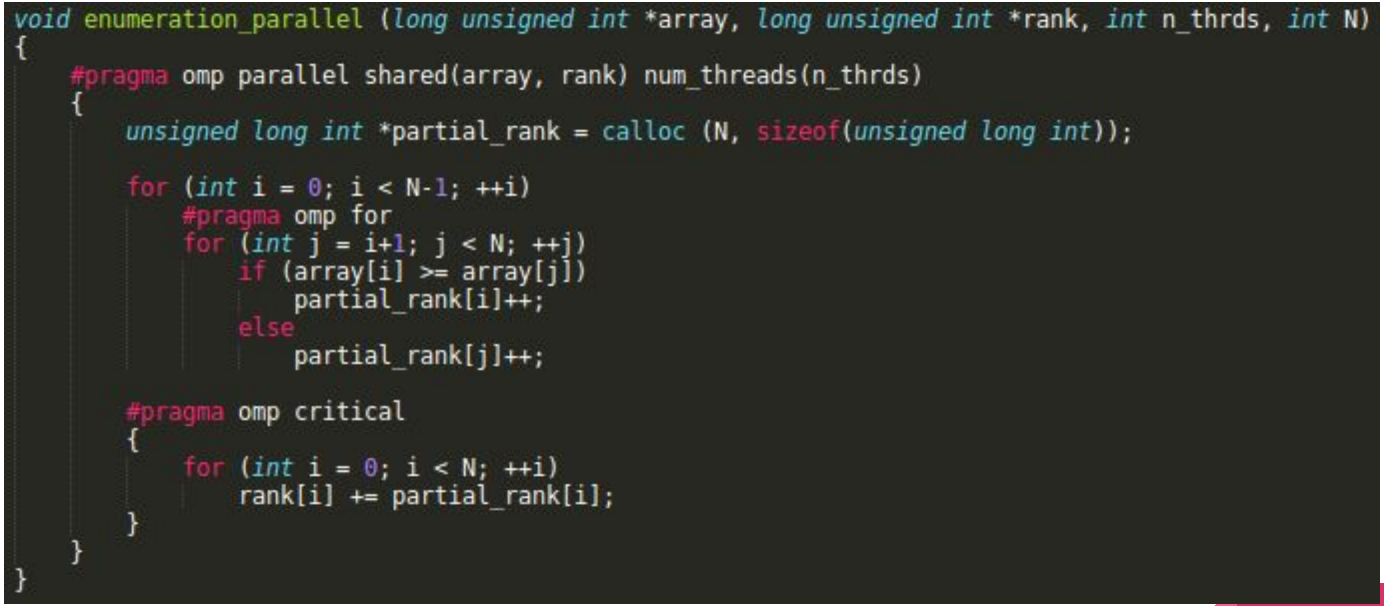
10	+1	+1	+1	
5	+1			
20	+1	+1	+1	+1
2				
8	+1	+1		

Código sequencial



```
void enumeration_sequencial (long unsigned int *array, long unsigned int *rank, int N)
{
    for (int i = 0; i < N-1; ++i)
        for (int j = i+1; j < N; ++j)
            if (array[i] >= array[j])
                rank[i]++;
            else
                rank[j]++;
}
```

Código paralelo



```c
void enumeration_parallel (long unsigned int *array, long unsigned int *rank, int n_thrds, int N)
{
    #pragma omp parallel shared(array, rank) num_threads(n_thrds)
    {
        unsigned long int *partial_rank = calloc (N, sizeof(unsigned long int));

        for (int i = 0; i < N-1; ++i)
            #pragma omp for
            for (int j = i+1; j < N; ++j)
                if (array[i] >= array[j])
                    partial_rank[i]++;
                else
                    partial_rank[j]++;

        #pragma omp critical
        {
            for (int i = 0; i < N; ++i)
                rank[i] += partial_rank[i];
        }
    }
}
```

PRAM

	$P(n)$	$T(n)$
CREW	$O(n^2)$	$O(\log n)$
CRCW	$O(n^2)$	$O(1)$

Tempo sequencial: $O(n^2)$

Amdahl

$$S(p) = \frac{1}{\beta + \frac{1-\beta}{p}} = S(p) = \frac{1}{0.01 + \frac{0.99}{p}}$$

$$S(2) = \frac{1}{0.01 + \frac{0.99}{2}} \approx 1.98$$

$$S(4) = \frac{1}{0.01 + \frac{0.99}{4}} \approx 3.88$$

$$S(8) = \frac{1}{0.01 + \frac{0.99}{8}} \approx 7.47$$

$$S(16) = \frac{1}{0.01 + \frac{0.99}{16}} \approx 13.91$$

$$S(\infty) = \frac{1}{0.01 + \frac{0.99}{\infty}} \approx \frac{1}{0.01} \approx 100$$

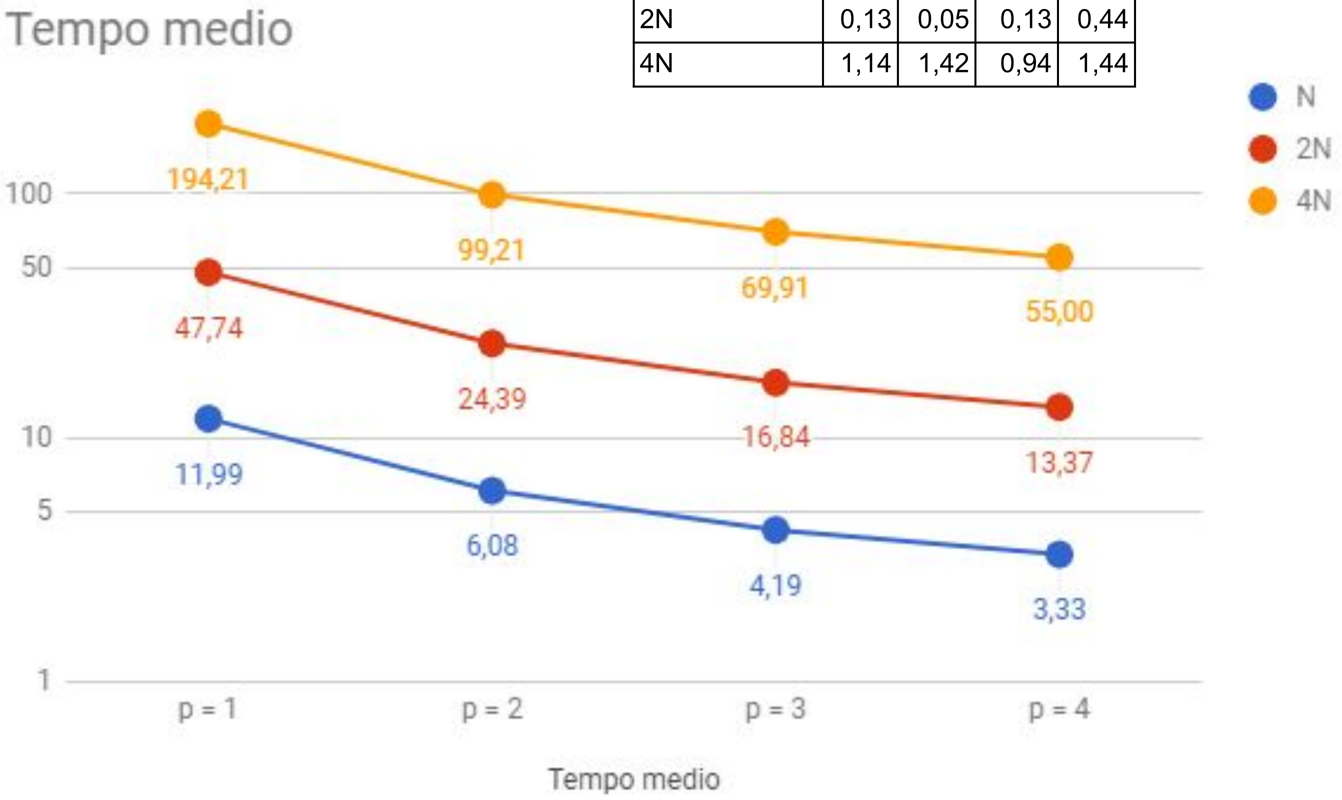
Gustafson-Barsis

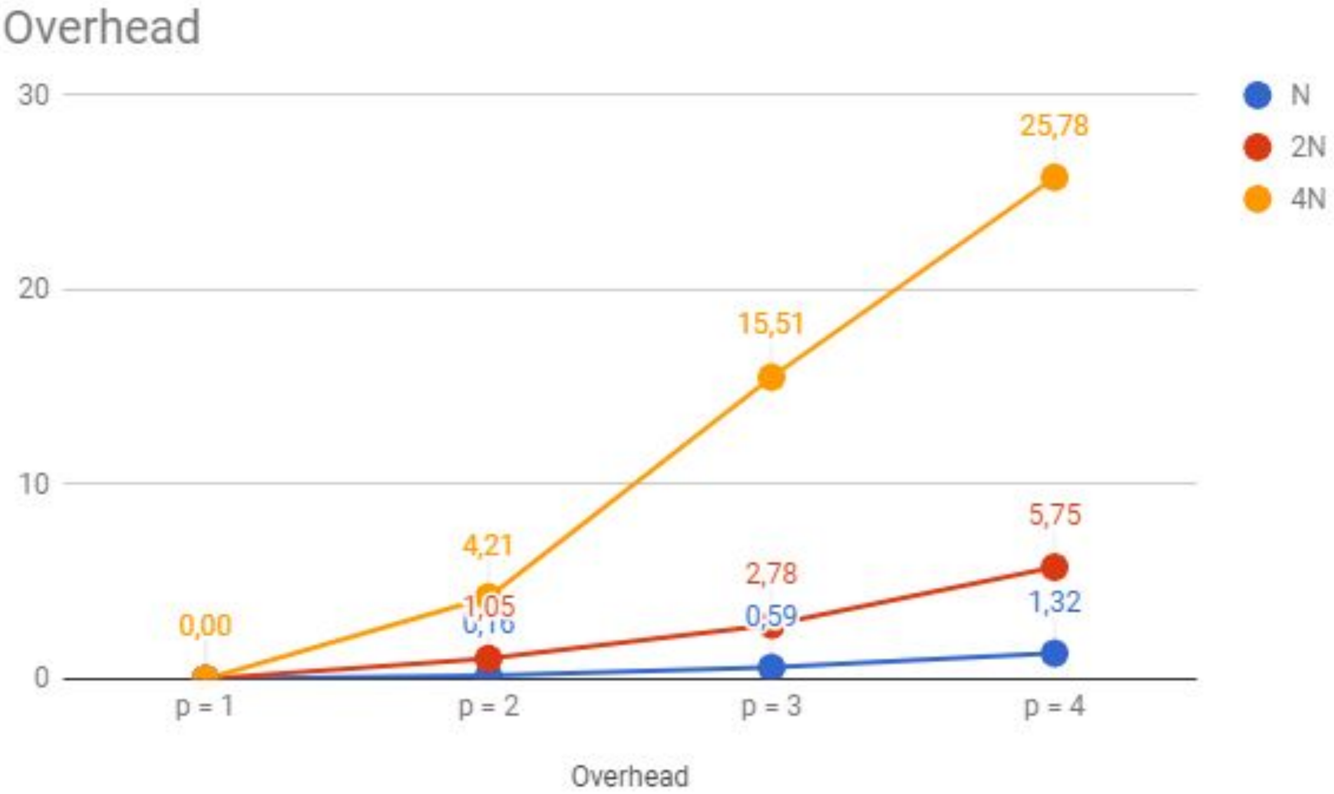
$$S(p) = \alpha + p \times (1 - \alpha)$$

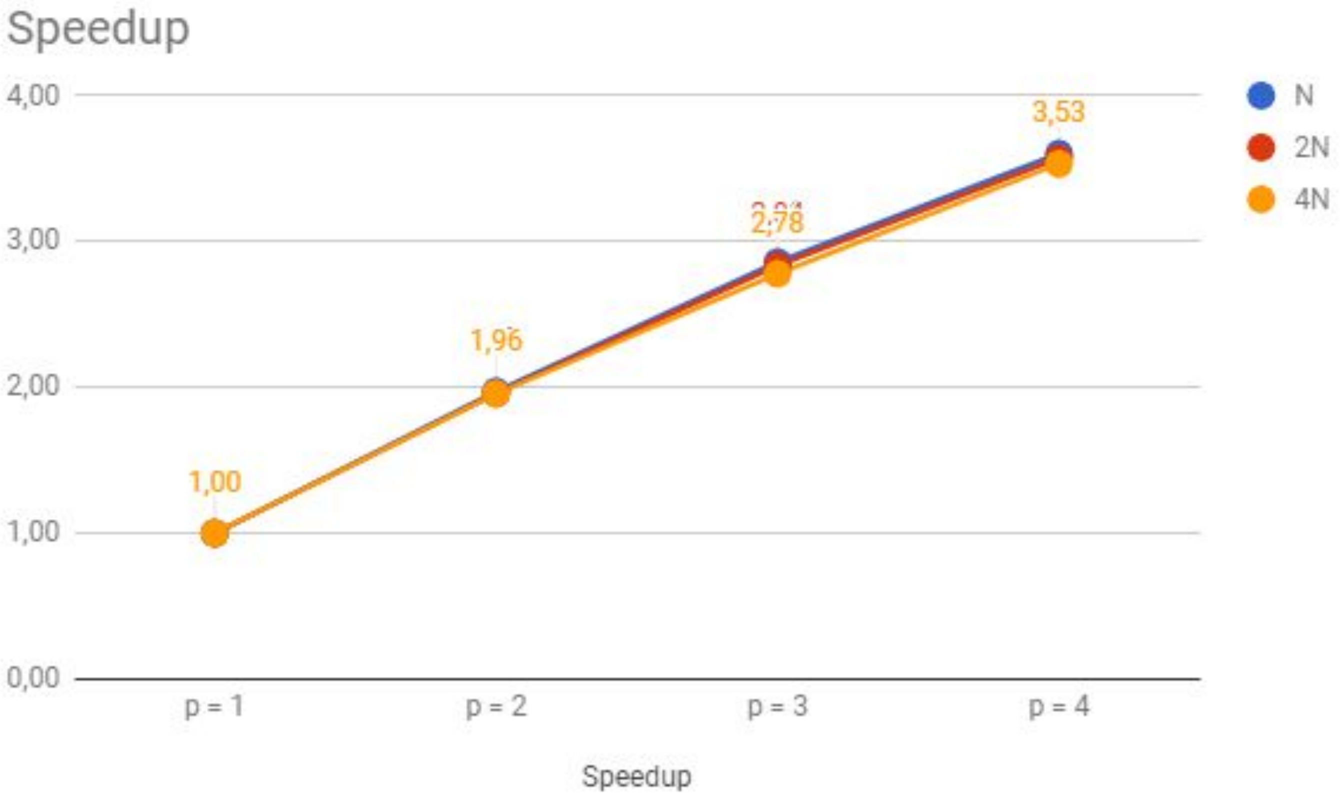
$$S(p) = 0.01 + p \times (0.99)$$

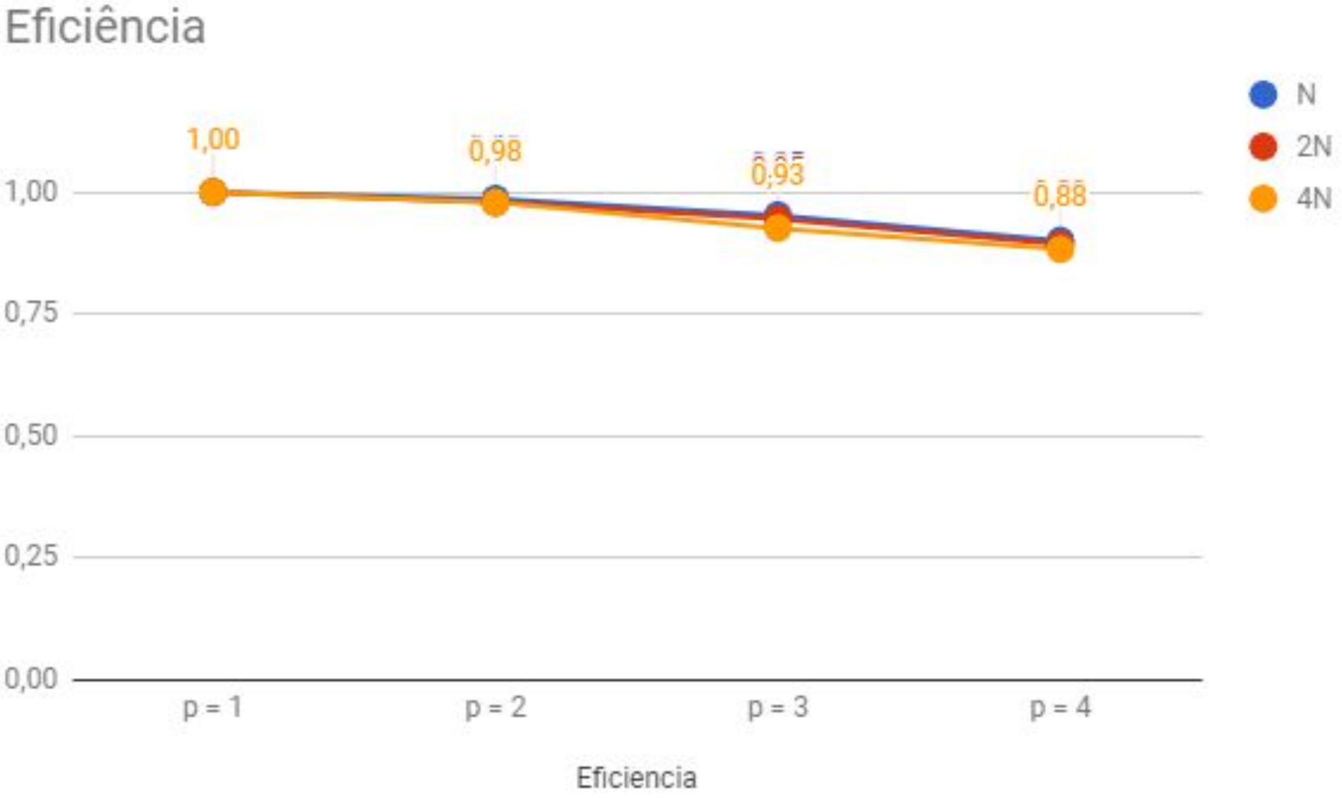
$$S(4) = 0.01 + 4 \times (0.99) \approx 3.97$$

Desvio Padrão	p = 1	p = 2	p = 3	p = 4
N	0,19	0,02	0,05	0,12
2N	0,13	0,05	0,13	0,44
4N	1,14	1,42	0,94	1,44




Overhead
30
20
10
0
N
2N
4N
25,78
15,51
4,21
0,00
5,75
2,78
1,05
0,16
0,59
1,32
p = 1
p = 2
p = 3
p = 4
Overhead


Speedup
4,00
3,00
2,00
1,00
0,00
1,00
1,96
2,78
3,53
p = 1
p = 2
p = 3
p = 4
Speedup
N
2N
4N


Eficiência
N
2N
4N
1,00
0,98
0,93
0,88
1,00
0,75
0,50
0,25
0,00
p = 1
p = 2
p = 3
p = 4
Eficiencia