

DrillComparingOrdersOfGrowth-1

Tom Bohbot

September 2020

1 Ordering Running Times

1.1 n^2

1. Double the input size:

- 1) When the input size, x , is doubled it will take this program $(2x)^2$ to run.
- 2) $(2x)^2 = 4x^2$
- 3) grows by a factor of 4

2. Increase the input size by one:

- 1) When the input size, x , is increased by one it will take this program $(x + 1)^2$ to run.
- 2) $(x + 1)^2 = x^2 + 2x + 1$
- 3) grows by a factor of $2x + 1$

1.2 n^3

1. Double the input size:

- 1) When the input size, x , is doubled it will take this program $(2x)^3$ to run.
- 2) $(2x)^3 = 8x^3$
- 3) grows by a factor of 8

2. Increase the input size by one:

- 1) When the input size, x , is increased by one it will take this program $(x + 1)^3$ to run.
- 2) $(x + 1)^3 = x^3 + 3x^2 + 3x + 1$
- 3) grows by a factor of $3x^2 + 3x + 1$

1.3 $100n^2$

1. Double the input size:

- 1) When the input size, x , is doubled it will take this program $(100(2x))^2$ to run.
- 2) $100(2x)^2 = (100)(4x)^2 = 400x^2$
- 3) $400x^2 / 100x^2 = 4$

4) grows by a factor of 4

2. Increase the input size by one:

- 1) When the input size, x , is increased by one it will take $100(x + 1)^2$ to run.
- 2) $100(x + 1)^2 = 100(x^2 + 2x + 1) = (100x^2 + 200x + 100)$
- 3) grows by a factor of $200x + 100$

1.4 $n \log n$

1. Double the input size:

- 1) When the input size, x , is doubled it will take this program $2x \log 2x$ to run.
- 2) $2x \log 2x = 2x \log x + 2x \log 2$ 3) grows by a factor of $2x \log x + 2x \log 2$

2. Increase the input size by one:

- 1) When the input size, x , is increased by one it will take $(x + 1) \log(x + 1)$ to run.
- 2) $(x + 1) \log(x + 1) = (x + 1) \log(x) + (x + 1) \log(1)$
- 3) grows by a factor of $(x + 1) \log(x) + (x + 1) \log(1)$

1.5 2^n

1. Double the input size:

- 1) When the input size, x , is doubled it will take this program $(2)^{2x}$ to run.
- 2) $(2)^{2x} = (2)^x * (2)^x$
- 3) grows by a multiplicative factor of $(2)^x$

2. Increase the input size by one:

- 1) When the input size, x , is increased by one it will take this program 2^{n+1} to run.
- 2) $2^{n+1} = 2^n * 2^1 = 2^n * 2$
- 3) grows by a multiplicative factor of 2

2 Really Understanding Order-of-Growth

2.1 n^2

Largest Input Size n to compute results in an hour assuming a computer that can do 10^{10} operations per second:

- 1) Maximum operations per minute = $60(10^{10}) = 600,000,000,000$ or 600 billion.
- 2) Maximum operations per hour = $60 \times$ maximum operations per minute = 36,000,000,000,000 or 36 trillion.
- 3) $n^2 = 36$ trillion

- 4) $n = 6,000,000$ or 6 million.
- 5) The maximum input size is 6,000,000.

2.2 n^3

Largest Input Size n to compute results in an hour assuming a computer that can do 10^{10} operations per second:

- 1) Maximum operations per minute = $60(10^{10} = 600,000,000,000$ or 600 billion.
- 2) Maximum operations per hour = $60 \times$ maximum operations per minute = 36,000,000,000,000 or 36 trillion.
- 3) $n^3 = 36$ trillion
- 4) $n = 33,019.3$
- 5) The maximum input size is 33,019.

2.3 $100n^2$

Largest Input Size n to compute results in an hour assuming a computer that can do 10^{10} operations per second:

- 1) Maximum operations per minute = $60(10^{10} = 600,000,000,000$ or 600 billion.
- 2) Maximum operations per hour = $60 \times$ maximum operations per minute = 36,000,000,000,000 or 36 trillion.
- 3) $100n^2 = 36$ trillion
- 4) $100n = 6,000,000$
- 5) $n = 60,000$
- 5) The maximum input size is 60,000.

2.4 $n \log n$

Largest Input Size n to compute results in an hour assuming a computer that can do 10^{10} operations per second:

- 1) Maximum operations per minute = $60(10^{10} = 600,000,000,000$ or 600 billion.
- 2) Maximum operations per hour = $60 \times$ maximum operations per minute = 36,000,000,000,000 or 36 trillion.
- 3) $n \log n = 36$ trillion
- 4) Through plugging and guessing, $(2,889,069,820,989) \log(2,889,069,820,989) = (2,889,069,820,989) * 12.46075804 \approx 35.99$ trillion. Any number larger than 2,889,069,820,989 is larger than 36 trillion.
- 5) Therefore, the maximum input size is 2,889,069,820,989.

NOTE: This is assuming log base 10.

Source for calculator: <https://dqydj.com/log-base-10-calculator/>

2.5 2^n

Largest Input Size n to compute results in an hour assuming a computer that can do 10^{10} operations per second:

- 1) Maximum operations per minute = $60(10^{10} = 600,000,000,000$ or 600 billion.
- 2) Maximum operations per hour = $60 \times$ maximum operations per minute = 36,000,000,000,000 or 36 trillion.
- 3) $2^n = 36$ trillion
- 4) Through plugging and guessing, 2^{45} is approximately 35.18 trillion. Therefore, n cannot exceed 45 as it would then be much higher than 35.18 trillion.
- 5) The maximum input size is 45.

2.6 2^{2^n}

Largest Input Size n to compute results in an hour assuming a computer that can do 10^{10} operations per second:

- 1) Maximum operations per minute = $60(10^{10} = 600,000,000,000$ or 600 billion.
- 2) Maximum operations per hour = $60 \times$ maximum operations per minute = 36,000,000,000,000 or 36 trillion.
- 4) Since we know that the largest input size for 2^n is 45, we only need to calculate for $2n = 45$.
- 5) $2n = 45$
- 6) $n = 22.5$
- 7) Let $n = 22$ as n must be a whole number.
- 8) $2^2 * 22$ is approximately 17.6 trillion
- 9) If $2^2 * 23$ was plugged in, then the result we be much higher than 36 trillion. Therefore, n cannot exceed 22
- 10) The maximum input size is 22.