計算量を学ぼう！

ぱうえる（けんた）

2022/12/20 ゼロイチゼミ @nu_zero_one

速いコードが書きたい！

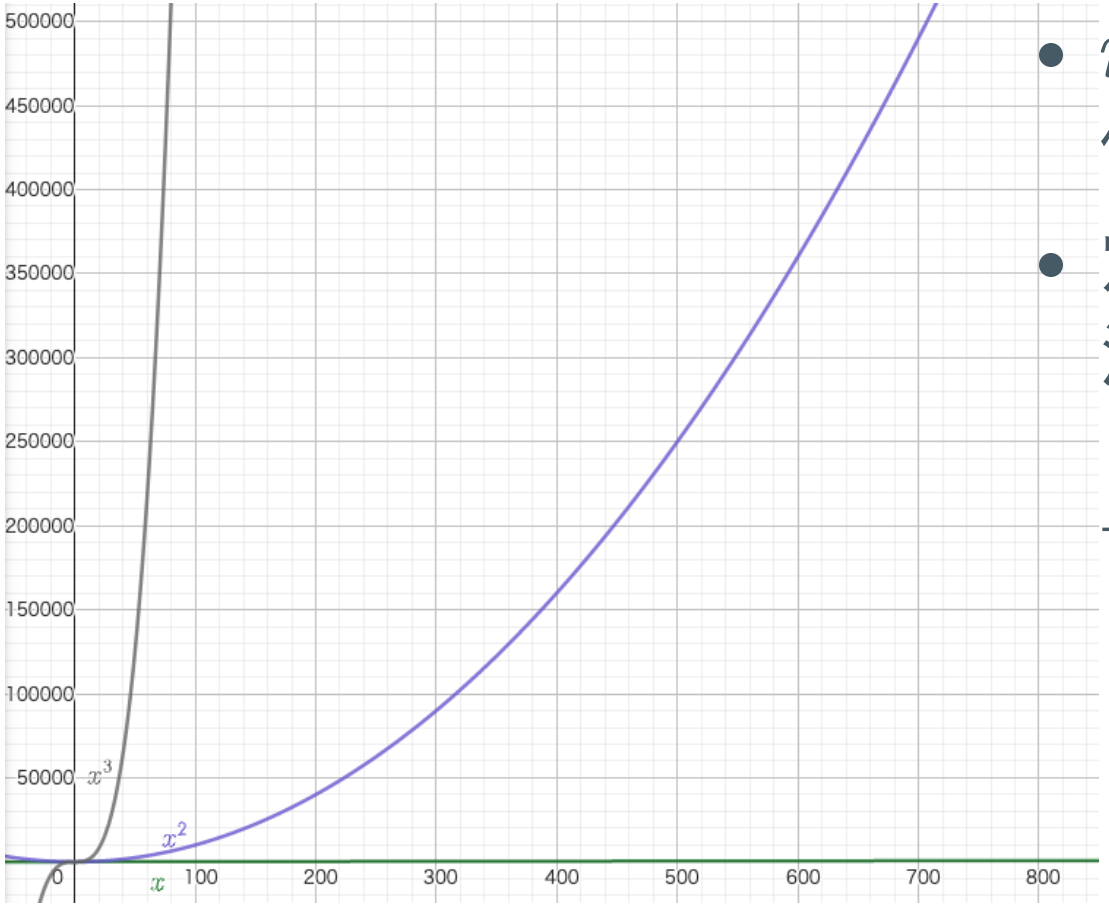
でも**速いコード**ってどうやって評価する？？

- 「$1,000,000$個のデータに対して5秒で終了しました！」
  - データの個数が変わったらどうなる？？
  - そもそもPythonで実行するかC言語で実行するかでも変わりそう

「データの大きさ」や「実行する環境」に依存しない評価方法が必要

→計算量の出番

オーダー記法 (1/2)



- n, n^2, n^3ではnが大きくなったとき値が大きく変化する
- 定数倍を考えないで、n の項だけに注目すればいいのでは？？

→ O (ランダウの記号)を用いて記述

オーダー記法 (2/2)

- 計算量は基本的にオーダー記法で書く
  1. 一番大きい項のみ残して表記する
     $n! > a^n > n^a > \log n > a$　（a は定数）
  2. 定数倍は無視する

オーダー記法の例）

$$5n^3 + 4n^2 + 100n \longrightarrow O(n^3)$$
$$2^n + n^{100} + 10^9 n \longrightarrow O(2^n)$$

コードの計算量の調べ方

- n 回のループをする $\rightarrow O(n)$
- n 回のループの中で n 回のループをする（二重ループ）
  $\rightarrow O(n^2)$
- bit全探索（n 個の要素について**ある/ない**の2通りを考える）
  $\rightarrow O(2^n)$
- n 個の順列を全て調べる $\rightarrow O(n!)$

問題！

このコードの計算量は？？（わかった人は高速化してみよう）

```python
# 1~n までの数の和を求める
n = int(input())

ans = 0
for i in range(1, n+1):
    ans += i

print(ans)
```

答え

このコードの計算量は？？（わかった人は高速化してみよう）

```python
# 1~n までの数の和を求める
n = int(input())

ans = 0
for i in range(1, n+1):
    ans += i

print(ans)
```

→ $O(n)$ （n までのループを1回している）

計算量を落とす (1/3)

上のコードは、$1 \sim n$ の和を求めるために $O(n)$ の計算をしています
（$n = 100,000,000$ で2.6秒くらい必要）

```
In [6]: %%timeit
   ...: # 1~n までの数の和を求める
   ...: n = 100_000_000
   ...:
   ...: ans = 0
   ...: for i in range(1, n+1):
   ...:     ans += i
   ...:
2.61 s ± 3.82 ms per loop (mean ± std. dev. of 7 runs, 1 loop each)
```

計算量を落とす (2/3)

この公式を使えば、、

$$\sum_{i=1}^{n} = \frac{1}{2}n(n+1)$$

```python
# 1~n までの数の和を求める
n = int(input())
ans = n * (n + 1) // 2

print(ans)
```

計算量を落とす (3/3)

```
In [9]: %%timeit
   ...: # 1~n までの数の和を求める
   ...: n = 100_000_000
   ...: ans = n * (n + 1) // 2
   ...:
   ...:
53.7 ns ± 3.87 ns per loop (mean ± std. dev. of 7 runs, 10,000,000 loops each)
```

なんと、53.7**ナノ**秒で終了！！

→ 約**5億倍**の高速化

参考

- 計算量オーダーの求め方を総整理！ ～どこからlogが出て来るか～