計数工学プログラミング演習　レポート課題

03-210634 木下裕太

2021 年 6 月 7 日

1　目的

疎行列の 2 乗を計算し、実行時間を測定する。実装方法の違いによって、速度にどのような差が生じるかを調べる。

2　方法

2 つの実装 Dense および Sparse に対して比較を行う。計算量についてはスカラーの四則演算が単位時間で行えると仮定。

2.1　Dense implementation

2 次元配列として行列を管理する。行列積は定義に基づいたシンプルな実装であり、疑似コードは以下のようになる。n を行列のサイズとして、時間計算量は $\Theta(n^3)$、空間計算量は $\Theta(n^2)$ である。

Algorithm 1 Dense version

Require:

A：$n \times n$ 行列

Ensure:

返り値は A^2。

```
function SquareDense(A)
    B ← O_n
    for i = 1, ··· , n do
        for k = 1, ··· , n do
            for j = 1, ··· , n do
                B_{i,j} ← B_{i,j} + A_{i,k} A_{k,j}
            end for
        end for
    end for
    return B
end function
```

2.2 Sparse implementation

全ての行と列について、含まれる非ゼロ成分を可変長リストで管理する。行列積の実装は先とほぼ同様だが、非ゼロ成分のみにアクセスする工夫をしている。疑似コードは以下の通りである。m を非ゼロ成分の個数とすると、時間計算量 $O(nm)$ で動作する。[*1]

Algorithm 2 Sparse version

```
1: function SquareSparse(A)
2:     B ← O_n
3:     for k = 1, ··· , n do
4:         for A_{i,k} ≠ 0 do
5:             for A_{k,j} ≠ 0 do
6:                 B_{i,j} ← B_{i,j} + A_{i,k} A_{k,j}
7:             end for
8:         end for
9:     end for
10:    return B
11: end function
```

2.3 実行環境

以下の実行環境で測定を行う。

CPU	Intel(R) Core(TM) i5-8250U, 1.60GHz
RAM	8 GB
OS	Windows 10 (64 bit)
Language	C (GCC 10.3.0)

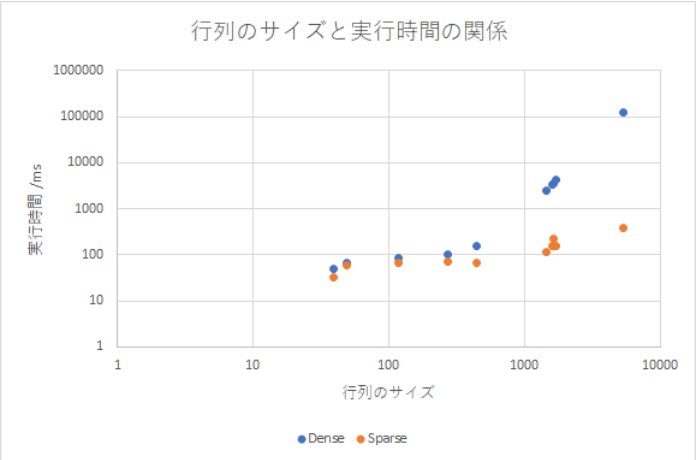
2.4 サンプルケース

Matrix Market のデータを入力として用いる。今回は疎行列のみを対象として測定を行う。各入力について 10 回繰り返し実行させ、所要時間の平均をその入力に対する実行速度とする。

[*1] 各非ゼロ成分は高々 2 回参照される。

3　実験結果

縦軸、横軸ともに対数スケールである。なお 2 つのコードの出力が全てのケースで一致することを確認した。



4　考察

予想される通り、$m \in o(n^2)$ である入力に対して Sparse の方がオーダーレベルで高速だった。今回の実装は一般に 2 つの疎行列の積を計算する場合にもそのまま同様に適用できるが、一方で 2 つの行列が同じであるという特殊な設定を上手く活かすことができなかった。