q1

October 1, 2020

[3]:
```
import numpy as np
import sys
import time

np.random.seed(24787)
a = np.random.randint(8, size=(3, 4, 4)) #(depth, rows, columns)
print('a= \n', a)
print('shape: ', a.shape)
```

```
a=
 [[[2 6 4 1]
  [0 4 4 3]
  [6 6 1 2]
  [7 0 6 5]]

 [[1 3 3 7]
  [4 7 2 5]
  [0 4 6 7]
  [5 5 7 1]]

 [[7 2 4 5]
  [6 7 7 0]
  [6 2 0 4]
  [2 0 7 6]]]
shape:  (3, 4, 4)
```

[4]:
```
b = np.tile(a, (1, 2, 2))
print('b= \n', b)
print('shape: ', b.shape)
```

```
b=
 [[[2 6 4 1 2 6 4 1]
  [0 4 4 3 0 4 4 3]
  [6 6 1 2 6 6 1 2]
  [7 0 6 5 7 0 6 5]
  [2 6 4 1 2 6 4 1]
  [0 4 4 3 0 4 4 3]
  [6 6 1 2 6 6 1 2]
```

```
  [7 0 6 5 7 0 6 5]]

 [[1 3 3 7 1 3 3 7]
  [4 7 2 5 4 7 2 5]
  [0 4 6 7 0 4 6 7]
  [5 5 7 1 5 5 7 1]
  [1 3 3 7 1 3 3 7]
  [4 7 2 5 4 7 2 5]
  [0 4 6 7 0 4 6 7]
  [5 5 7 1 5 5 7 1]]

 [[7 2 4 5 7 2 4 5]
  [6 7 7 0 6 7 7 0]
  [6 2 0 4 6 2 0 4]
  [2 0 7 6 2 0 7 6]
  [7 2 4 5 7 2 4 5]
  [6 7 7 0 6 7 7 0]
  [6 2 0 4 6 2 0 4]
  [2 0 7 6 2 0 7 6]]]
shape:  (3, 8, 8)
```

[6]:
```
c = []
for dpth in b:
    sum_rw = 0
    for rw in dpth:
        sum_rw += np.sum(rw)
        # print(sum_rw)
    c.append(sum_rw)
    # print(sum_rw)
c = np.array(c)
print('c= ', c)
```

```
c=  [228 268 260]
```

[7]:
```
np.random.seed(24787)
a = np.random.randint(low=0, high=8, size=(1000,1000))
b = np.random.randint(low=0, high=8, size=(1000,1000))



def matmul(a, b):
    #performs a matrix multiplication taking dot product manually...
    a_shape = a.shape
    b_shape = b.shape
    mult = np.empty([a_shape[0], b_shape[1]])
    if a_shape[1] != b_shape[0]:
        sys.exit('size not compatible')
```

```
    else:
        for i in range(len(a)): #row
            for j in range(len(b[0])): #column
                for k in range(len(b)):
                    mult[i][j] += a[i][k] * b[k][j]

    return mult
start_time = time.time()
print(matmul(a, b))

print("%s seconds" %(time.time() - start_time))
```

```
[[12146. 12253. 12302. … 12123. 12415. 12239.]
 [12251. 12131. 12180. … 12691. 12396. 12497.]
 [11434. 11864. 12043. … 12348. 11960. 12207.]
 …
 [11774. 11945. 12276. … 12339. 12178. 12059.]
 [11627. 12167. 12254. … 11929. 11958. 12078.]
 [11560. 12145. 12077. … 12210. 12124. 12031.]]
1562.5061054229736 seconds
```

[8]:
```
start_time = time.time()
print(a@b)
print("%s seconds" %(time.time() - start_time))
```

```
[[12146 12253 12302 … 12123 12415 12239]
 [12251 12131 12180 … 12691 12396 12497]
 [11434 11864 12043 … 12348 11960 12207]
 …
 [11774 11945 12276 … 12339 12178 12059]
 [11627 12167 12254 … 11929 11958 12078]
 [11560 12145 12077 … 12210 12124 12031]]
2.539484739303589 seconds
```

The matrix multiplication symbol has smaller memory consumption than the matmul function, which is why it is faster

[ ]: