程式作業三

107502540 陳皇宇

Dynamic Programming

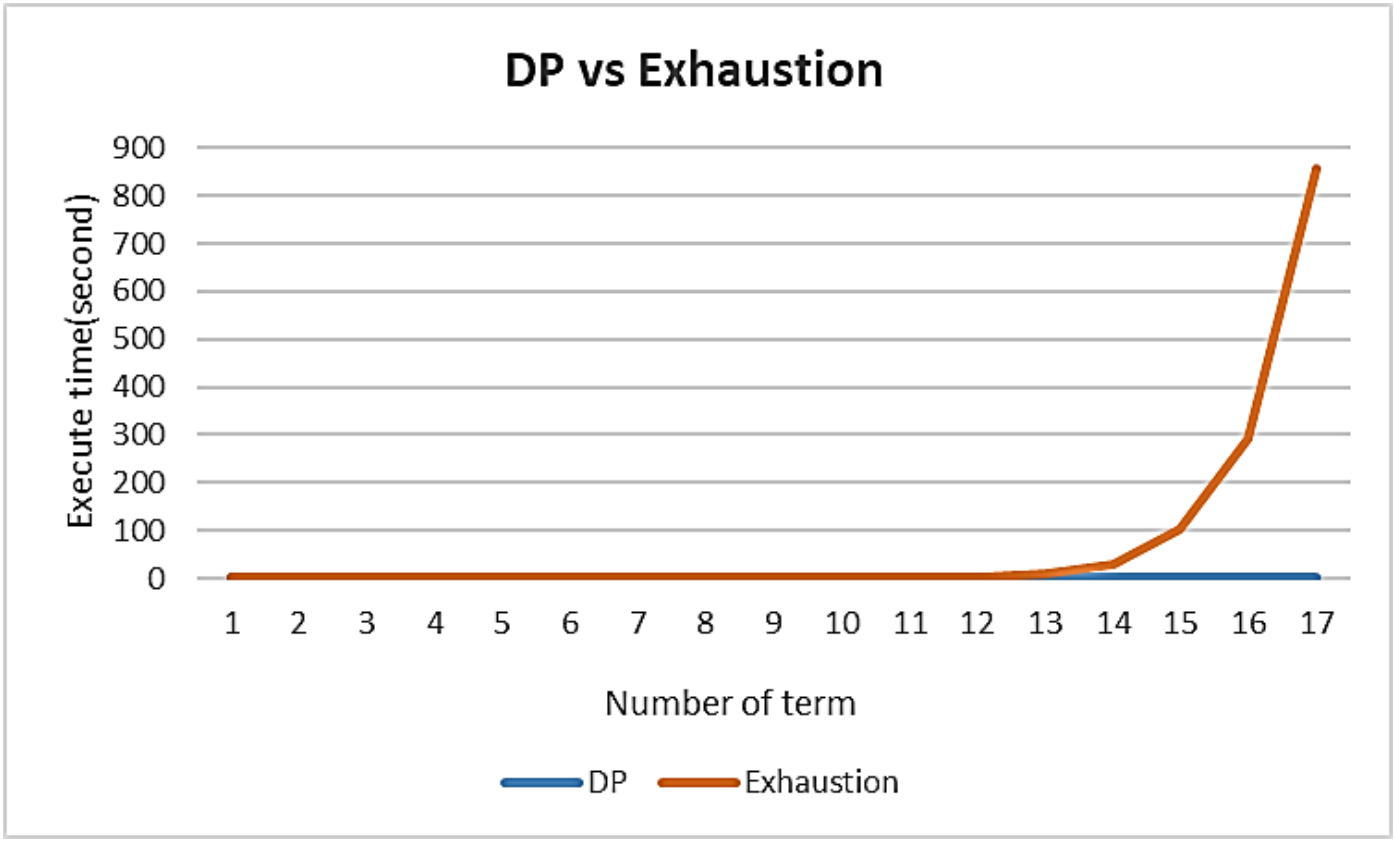
資料結構：用一個 2 dim int array 跟一個 2 dim string array

演算法：動態規劃

時間複雜度：O(n^3)

Pseudo code：

```
def DP(numberList, opList):
    #初始化
    for i 0 to len(numberList)
        dp[i][i] = numberList[i]
        step[i][i] = str(numberList[i])
    for i =1 to len(numberList)
        for j =0 to len(numberList)-i
            for k=0 to i
                op = opList[k+j]
                num1 = dp[j][j+k]
                num2 = dp[j+k+1][i+j]
                num2_min = dp[i+j][j+k+1]
                value = (num1+num2) if op == "+" else ((num1-num2) if op == "-" else
(num2 * num1))
                valuemin = (num1+num2_min) if op == "+" else ((num1-num2_min) op
== "-" else (num2_min * num1))
                maxvalue = max(value, valuemin)
                minvalue = min(value, valuemin)
                if dp[j][i+j] < maxvalue:
                    dp[j][j+i] = maxvalue
                    step[j][j+i] = "("+step[j][j+k]+op+step[j+k+1][i+j]+")"
                if dp[i+j][j] > minvalue:
                    dp[i+j][j] = minvalue
                    step[i+j][j] = "("+step[j][j+k]+op+step[j+k+1][i+j]+")"
return dp[0][len(numberList)-1], step[0][len(numberList)-1][1:-1]
```


DP vs Exhaustion
Execute time(second)
900
800
700
600
500
400
300
200
100
0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
Number of term
DP
Exhaustion