Algorithm PA2 alg20161 B07901187 宋家齊

Run time and memory usage:

12: 3

```
bash-4.1$ time ./bin/mps inputs/12.in outputs/12.out
The total CPU time: 4.998ms
memory: 12460KB

real    0m0.056s
user    0m0.003s
sys     0m0.006s
```

1000: 52

```
bash-4.1$ time ./bin/mps inputs/1000.in outputs/1000.out
The total CPU time: 24.995ms
memory: 14572KB

real    0m0.038s
user    0m0.022s
sys     0m0.007s
```

10000: 176

```
bash-4.1$ time ./bin/mps inputs/10000.in outputs/10000.out
The total CPU time: 2215.66ms
memory: 220928KB

real    0m2.236s
user    0m2.060s
sys     0m0.171s
```

100000: 566

```
bash-4.1$ time ./bin/mps inputs/100000.in outputs/100000.out
The total CPU time: 198041ms
memory: 20818564KB

real    4m32.593s
user    3m3.749s
sys     0m14.748s
```

Data structure:

使用 n*n 的 vector<int>來存 maximum planar subset 的大小，判斷條件就和 HW2 一樣，建表利用 top down，並記錄已算過的值，使用 bottom up 會較慢，因為不見得每格都會用到。同時另建一個 n*n 的 vector<bool>，在有選取 k chord 時紀錄 1，若無則記錄 0。最後利用 top down 方式從最大的 subset 開始找要從哪裡分割，也就是是否有選取到 k chord。Top down 可使用 stack 來實作取代 recursive，否則會使用過多記憶體。

Algorithm PA2 alg20161 B07901187 宋家齊

Review:
基本上這份作業有兩個重點，一是建表，二是找出所有的 Chord set。
建表部分，一開始我分別寫 top down 和 bottom up 的做法，想說 top down 會耗非常多記憶體，而改用 bottom up，bottom up 在一萬的 case 跑的不錯，但十萬的 case 卻會不合理地花非常多時間，因此不得不修改。直到最後發現是我 top down 忘記利用 memorization，最後改成 top down 的方法 run time 改善很多。

紀錄 Chord set 部分，一開始我也是用 top down，很耗記憶體，接著我把每一格的 Chord subset 記錄下來，遇到 case2,3 就將 subset concatenate，記憶體會少一些，但對十萬的 case 來說還是太大。最後回到原本的想法，用 stack 實作 recursive function call。
在這份作業中深刻體會到 top down 的效率基本上>=bottom up。

感謝我的組員、stan 和田哥。

Improvement:
有些可能可以改善 Runtime 和 memory 的方法，例如把 M 和 S 表格寫成三角形那樣，像是 HW2 所畫的；或是把 int 改成 short int 等等；用 vector<Chord>來存取 data 而不用 map<int,int>因為 map 會使用較多記憶體。