雲端運算程式設計期中報告
第13組

8105056009陳原禾
7106056054黃筱真
7106056091李明翰
7106056111徐昀汝

1. SPARK 實作k-Means

Implement

- 創建RDD(pandas dataframe 轉成 RDD Row(Feature_0. Feature_1…)
- 隨機選取 k 個群心，當作初始化群心

```
'''initial center'''
K = 20
RDDdf = sc.parallelize(df.values)
#print(RDDdf.take(5))
#print(len(df.feature_1))
center = RDDdf.takeSample(False,K,1)
#print(center)
```

Implement

- 計算某個點的類別(被分到哪個群)
- 初始化某個點與任意一個群心的最小距離為無限大
- 更新與任意一個群心的最小距離
- 更新類別

```
def group(node,center):
    min_dist = 9999999
    group_idx = 0
    for i in range(len(center)):
        dist = np.sum(pow((node - center[i]),2))
        if dist < min_dist:
            min_dist = dist
            group_idx = i
    return group_idx
```

Implement

- Map：

  每一個RDD去計算所有點與群心的距離，算出離自己最近的群心，return(群心,(資料點))

```
group_result = RDDdf.map(lambda n: (group(n,center), (n,1) ) )
```

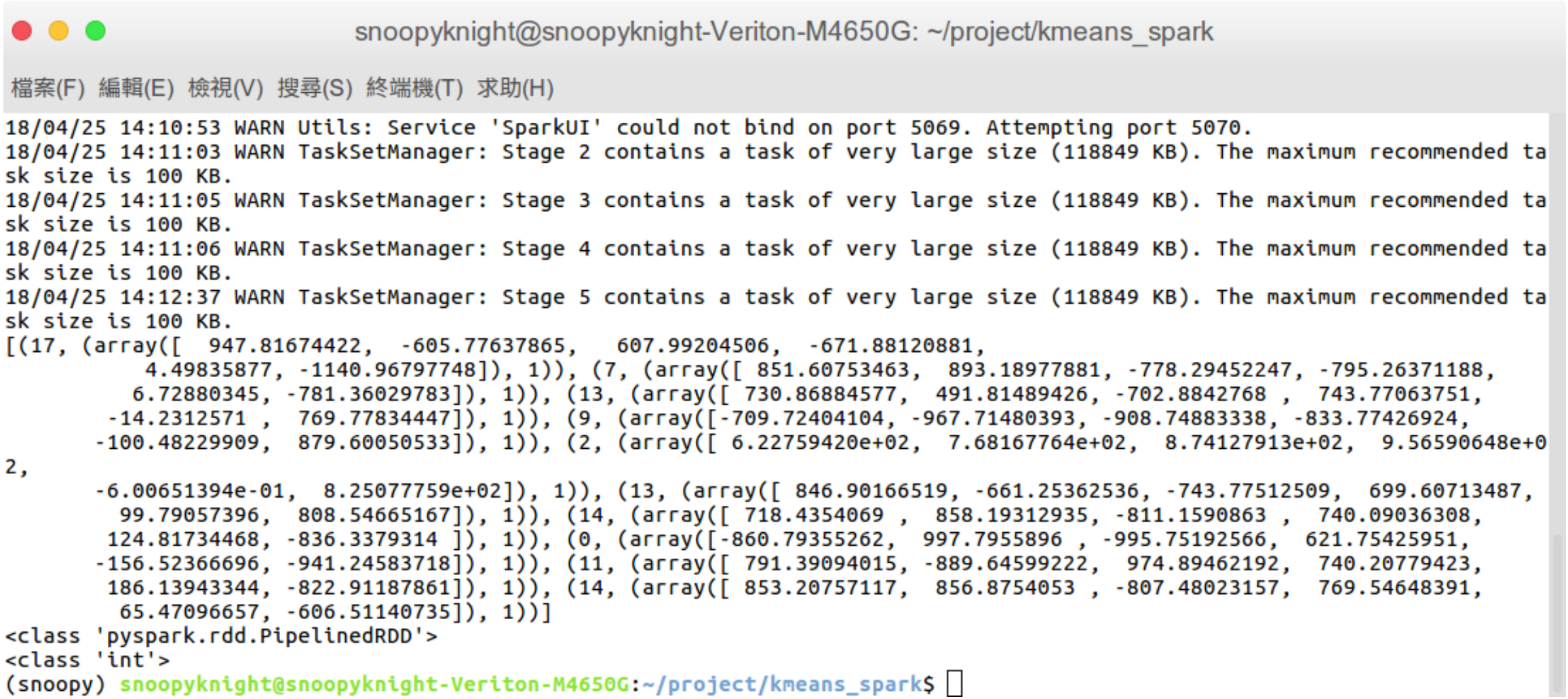
- Reduce by key：

  分群後，將某一群中的所有點加總，並計算某一群的總點，reduce完後新群心產出

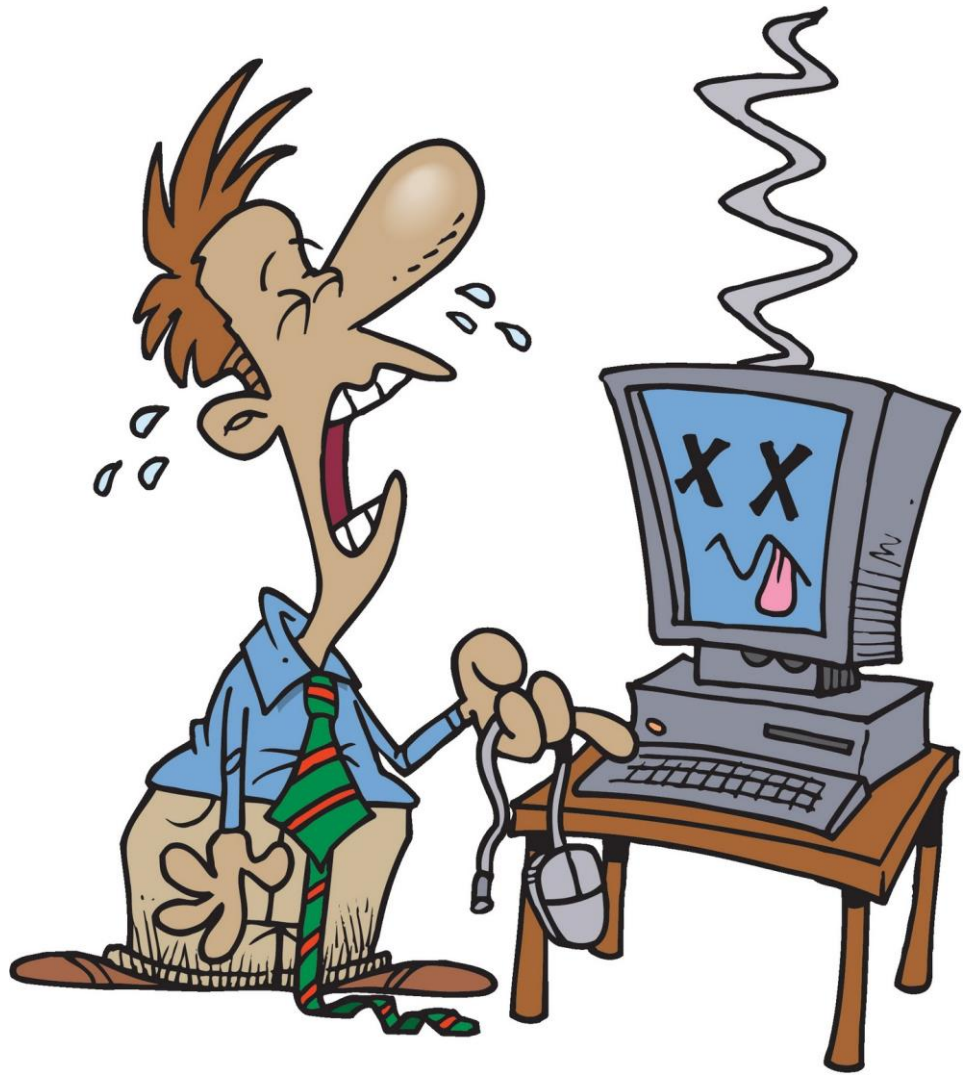
```
sum_node = group_result.reduceByKey(lambda x,y: x+y)
```

Problem

- Spark 邏輯與之前學的程式不同
- 轉換RDD的方式 :
  - pandas dataframe 以 sc.parallelize() 轉換成RDD
  - 使用 pyspark.sql dataframe
  - 以 sc.textfile 讀資料，轉換成RDD
- map, reduce 操作資料的方式不熟悉


snoopyknight@snoopyknight-Veriton-M4650G: ~/project/kmeans_spark
檔案(F) 編輯(E) 檢視(V) 搜尋(S) 終端機(T) 求助(H)
18/04/25 14:10:53 WARN Utils: Service 'SparkUI' could not bind on port 5069. Attempting port 5070.
18/04/25 14:11:03 WARN TaskSetManager: Stage 2 contains a task of very large size (118849 KB). The maximum recommended ta
sk size is 100 KB.
18/04/25 14:11:05 WARN TaskSetManager: Stage 3 contains a task of very large size (118849 KB). The maximum recommended ta
sk size is 100 KB.
18/04/25 14:11:06 WARN TaskSetManager: Stage 4 contains a task of very large size (118849 KB). The maximum recommended ta
sk size is 100 KB.
18/04/25 14:12:37 WARN TaskSetManager: Stage 5 contains a task of very large size (118849 KB). The maximum recommended ta
sk size is 100 KB.
[(17, (array([947.81674422, -605.77637865, 607.99204506, -671.88120881,
4.49835877, -1140.96797748]), 1)), (7, (array([851.60753463, 893.18977881, -778.29452247, -795.26371188,
6.72880345, -781.36029783]), 1)), (13, (array([730.86884577, 491.81489426, -702.8842768 , 743.77063751,
-14.2312571 , 769.77834447]), 1)), (9, (array([-709.72404104, -967.71480393, -908.74883338, -833.77426924,
-100.48229909, 879.60050533]), 1)), (2, (array([6.22759420e+02, 7.68167764e+02, 8.74127913e+02, 9.56590648e+0
2,
-6.00651394e-01, 8.25077759e+02]), 1)), (13, (array([846.90166519, -661.25362536, -743.77512509, 699.60713487,
99.79057396, 808.54665167]), 1)), (14, (array([718.4354069 , 858.19312935, -811.1590863 , 740.09036308,
124.81734468, -836.3379314]), 1)), (0, (array([-860.79355262, 997.7955896 , -995.75192566, 621.75425951,
-156.52366696, -941.24583718]), 1)), (11, (array([791.39094015, -889.64599222, 974.89462192, 740.20779423,
186.13943344, -822.91187861]), 1)), (14, (array([853.20757117, 856.8754053 , -807.48023157, 769.54648391,
65.47096657, -606.51140735]), 1))]
<class 'pyspark.rdd.PipelinedRDD'>
<class 'int'>
(snoopy) snoopyknight@snoopyknight-Veriton-M4650G:~/project/kmeans_spark$

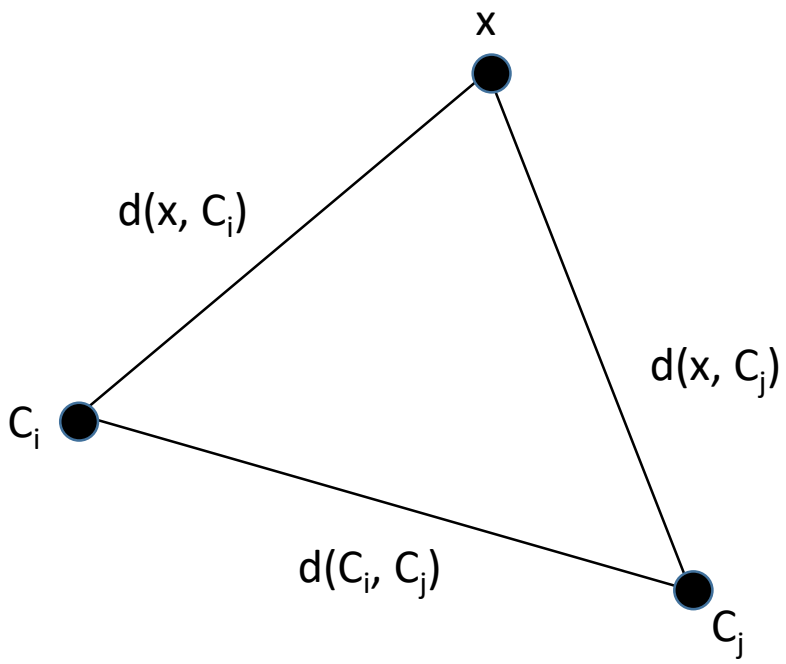
當我們要 update center 時....


X X

2. 運用三角不等式加速k-Means

三角不等式

- 任一三角形，兩邊之合必大於等於第三邊，兩邊之差必小於第三邊。
- 定義:
  資料點集合 : $X = \{x_1, x_2, \ldots, x_n\}$
  群心點集合 : $C = \{c_1, c_2, \ldots, c_n\}$
- 定理1:
  if $2d(x, C_i) \le d(C_i, C_j) \rightarrow d(x, C_i) \le d(x, C_j)$
- 定理2:
  設$C_i \in C, \exists C_j \in C$,使得$d(C_i, C_j) = \min d(C_i, C)$
  對於資料點$x \in X$,
  if $2d(x, C_i) \le d(C_i, C_j) \rightarrow \min d(x, C) = d(C_i, x)$



隨機選擇*k*個資料點作為初始質心

while:

計算*k*個質心間的距離，並且保存每個質心的到其他質心的最短距離。

for 對於每個數據點x :

if 資料點x已分配在質心C_i所在群:

if 2d(Ci, x) <= d(Ci, Cj)

x分配無需變動;

else:

繼續計算x到現有*k*個質心的距離，將它歸到距離最近質心的所在群中

else (資料點x未分配到任何群):

for i from 0 to K **do**

if 2d(C_i, x) <= d(C_i, C_j):

將x歸屬到Ci所在群中

退出for迴圈

重新計算群心

if 跑完10 round :

break

跑10-iterations.txt - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明(H)

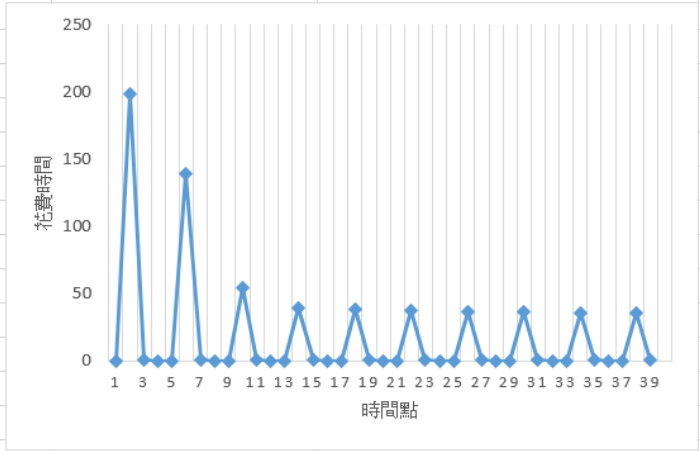
```
回合 1
1.回合開始: 0.1718883514404297
2.取得20個質心之間的最短距離完 0.18751120567321777
3.三角不等式判斷完: 198.96497440338135
4.更新質心完: 199.2462387084961
SSE = 825779451027.1733

回合 2
1.回合開始: 199.2462387084961
2.取得20個質心之間的最短距離完 199.2462387084961
3.三角不等式判斷完: 338.821170091629
4.更新質心完: 339.0711827278137
SSE = 643418749614.4862

回合 3
1.回合開始: 339.0711827278137
2.取得20個質心之間的最短距離完 339.0711827278137
3.三角不等式判斷完: 393.46641087532043
4.更新質心完: 393.732052564621
SSE = 617478601874.8972

回合 4
1.回合開始: 393.732052564621
2.取得20個質心之間的最短距離完 393.732052564621
3.三角不等式判斷完: 432.7465784549713
4.更新質心完: 433.0122218132019
SSE = 616973161628.3274

回合 5
1.回合開始: 433.0122218132019
2.取得20個質心之間的最短距離完 433.0122218132019
3.三角不等式判斷完: 471.1106605529785
4.更新質心完: 471.34505009651184
SSE = 616913162801.1018

回合 6
1.回合開始: 471.3606786727905
2.取得20個質心之間的最短距離完 471.3606786727905
3.三角不等式判斷完: 508.80417108535767
4.更新質心完: 509.085435628891
SSE = 616866756392.02

回合 6
1.回合開始: 471.3606786727905
2.取得20個質心之間的最短距離完 471.3606786727905
3.三角不等式判斷完: 508.80417108535767
4.更新質心完: 509.085435628891
SSE = 616866756392.02

回合 7
1.回合開始: 509.085435628891
2.取得20個質心之間的最短距離完 509.085435628891
3.三角不等式判斷完: 545.7288360595703
4.更新質心完: 545.9946534633636
SSE = 616834141491.917

回合 8
1.回合開始: 545.9946534633636
2.取得20個質心之間的最短距離完 545.9946534633636
3.三角不等式判斷完: 582.265734910965
4.更新質心完: 582.5313727855682
SSE = 616815462629.0178

回合 9
1.回合開始: 582.5313727855682
2.取得20個質心之間的最短距離完 582.5469992160797
3.三角不等式判斷完: 617.9577355384827
4.更新質心完: 618.2233748435974
SSE = 616805813513.8394

回合 10
1.回合開始: 618.2233748435974
2.取得20個質心之間的最短距離完 618.2389998435974
3.三角不等式判斷完: 653.6282994747162
4.更新質心完: 653.8627791404724
SSE = 616801748425.0045


分群結果: [19  8  5 ... 19 16  6]
SSE = 616801748425.0045
Time taken:  653.8627791404724 seconds.
收斂所需迭代次數: 10
```

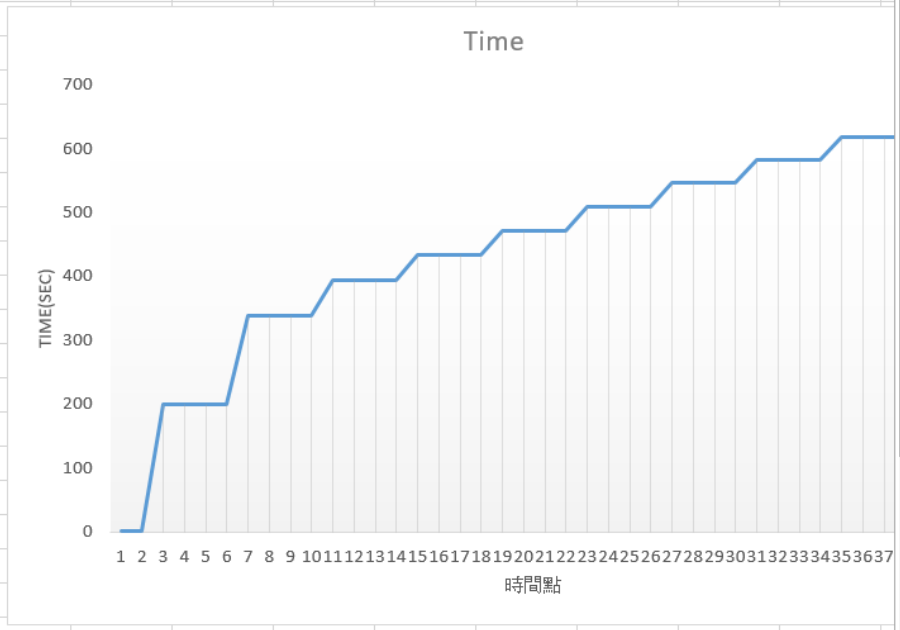
統計資料.xlsx [受保護的檢視] - Excel

檔案 常用 插入 版面配置 公式 資料 校閱 檢視 ACROBAT 小組 告訴我您想要執行的動作... 登入 共用

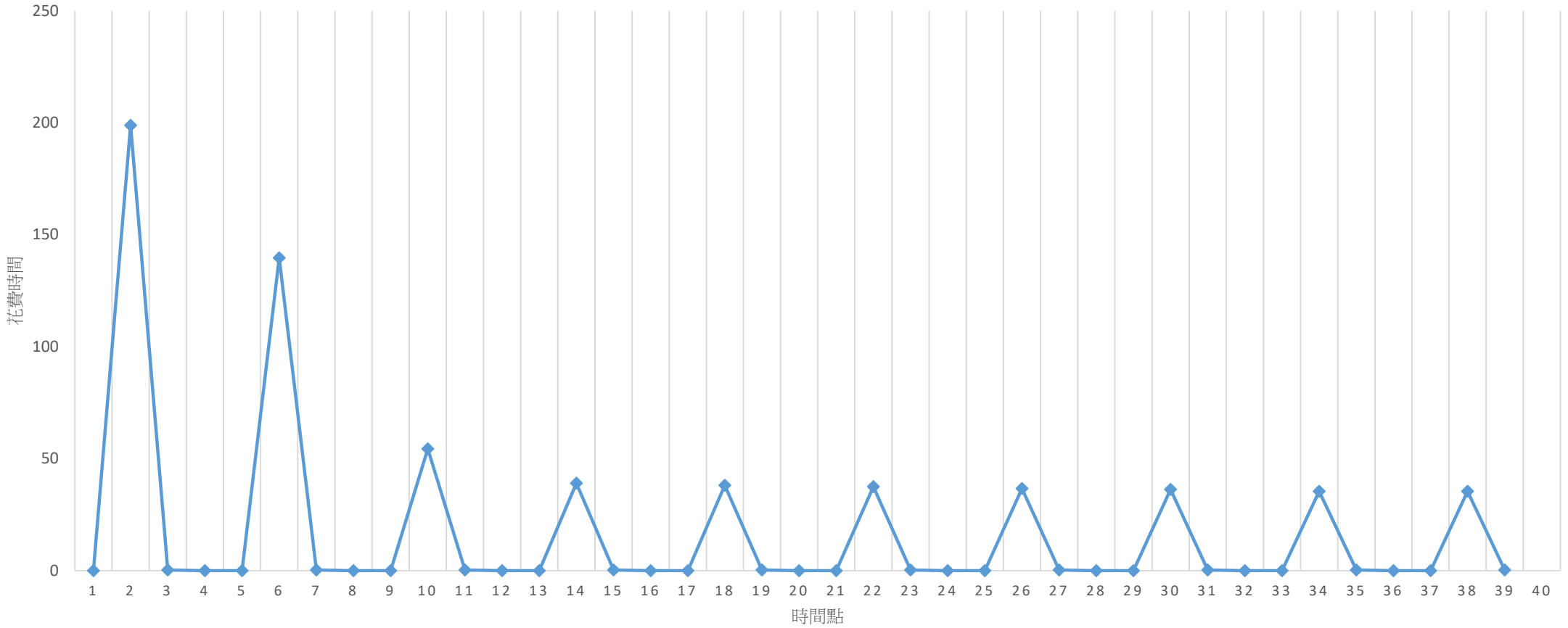
D12

	A	B	C	D	E
1	Timestamp	Timestamp差		SSE	SSE差
2	0.171888351	0.015622854		8.25779E+11	1.82361E+11
3	0.187511206	198.7774632		6.43419E+11	25940147740
4	198.9649744	0.281264305		6.17479E+11	505440246.6
5	199.2462387	0		6.16973E+11	59998827.23
6	199.2462387	0		6.16913E+11	46406409.08
7	199.2462387	139.5749314		6.16867E+11	32614900.1
8	338.8211701	0.250012636		6.16834E+11	18678862.9
9	339.0711827	0		6.16815E+11	9649115.178
10	339.0711827	0		6.16806E+11	4065088.835
11	339.0711827	54.39522815		6.16802E+11	6.16802E+11
12	393.4664109	0.265641689			0
13	393.7320526	0			
14	393.7320526	0			
15	393.7320526	39.01452589			
16	432.7465785	0.265643358			
17	433.0122218	0			
18	433.0122218	0			
19	433.0122218	38.09843874			
20	471.1106606	0.234389544			
21	471.3450501	0.015628576			
22	471.3606787	0			
23	471.3606787	37.44349241			
24	508.8041711	0.281264544			
25	509.0854356	0			
26	509.0854356	0			
27	509.0854356	36.64340043			
28	545.7288361	0.265817404			
29	545.9946535	0			
30	545.9946535	0			





每一 Round 所花的時間



The End
Thank You !