Digital System Design
FINAL : Hierarchical Clustering

105060016 謝承儒 電資 20

一. 目的

把散亂的測資中，較為接近的分成一群，如此就能將測資稍微整理一下。

此次的測資是 hw06 中圖片的第一個 pixel，因此這是把相近顏色的分成同一堆。

二. 方法

1. 將每筆測資視為獨立的群聚(cluster)，n 筆測資就有 n 個群聚。
2. 計算兩個群聚間的距離。

$$d(C_i, C_j) = \sum_{\mathbf{a} \in C_i, \mathbf{b} \in C_j} \frac{d(\mathbf{a}, \mathbf{b})}{|C_i||C_j|},$$

3. 找出最接近的兩個群聚，將他們相連在一起
4. 重複步驟 2.3，直到群聚數量剩下希望的數量。

三. 軟、硬體工作分配

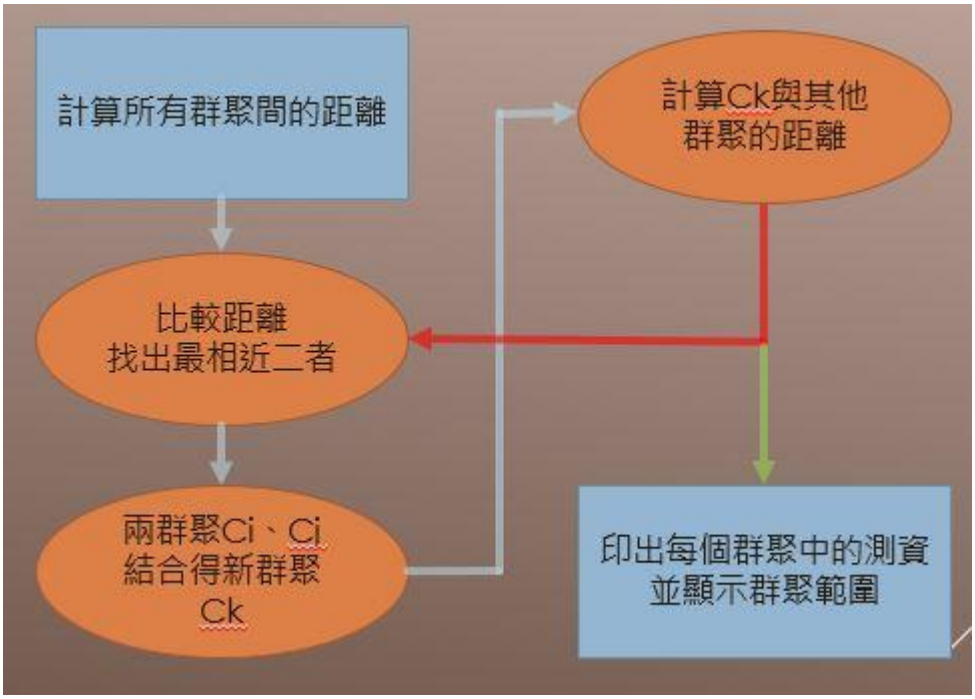
1. 軟體

   A. 找出最接近的兩群聚，並接上
   B. 尋找在該群聚的測資

2. 硬體

   A. 計算兩群聚間的距離
   B. 印出結果

四． 程式運行過程



五． 軟體中的變數

1. node_dis[NUM_TEST_IMAGE][NUM_TEST_IMAGE]
   兩測資間的距離
2. cluster_dis[NUM_TEST_IMAGE][NUM_TEST_IMAGE]
   兩群聚間的距離
3. cluster[NUM_TEST_IMAGE]
   每個測資所屬的群聚，起初每個都屬於自己(cluster[i]=i)，但一旦接上就會改變裡面的值，例如 3 和 50 接上就會將 cluster[50]=3，但 cluster[3]=3。
4. root_1, root_2, child_1, child_2,
5. cluster_num = NUM_TEST_IMAGE
   剩下的群聚數量。

六. 軟體中的設計

1. 尋找最接近兩群聚 Ci,Cj

裡用 cluster_dis 這個陣列，用雙重迴圈去倆倆比對。找出最接近的 Ci,Cj。

2. 將 Ci,Cj 合成為 Ck(Ci<Cj)

利用迴圈找出 cluster[i]==Cj 並改成 Ci，如此代表將兩群聚合在一起。

3. 計算 Ck(變大的 Ci)與其他群聚的距離

將 root1 設為 Ck，做以下步驟 ：

A. 利用迴圈，找出其他的群聚 Cw(不等於 Ci)，並把 root2 設為 Cw

B. 再利用第二層迴圈找出 cluster[i]==root1，將 child1 設為 i。

C. 再利用第三層迴圈找出 cluster[j]==root2，將 child2 設為 j。

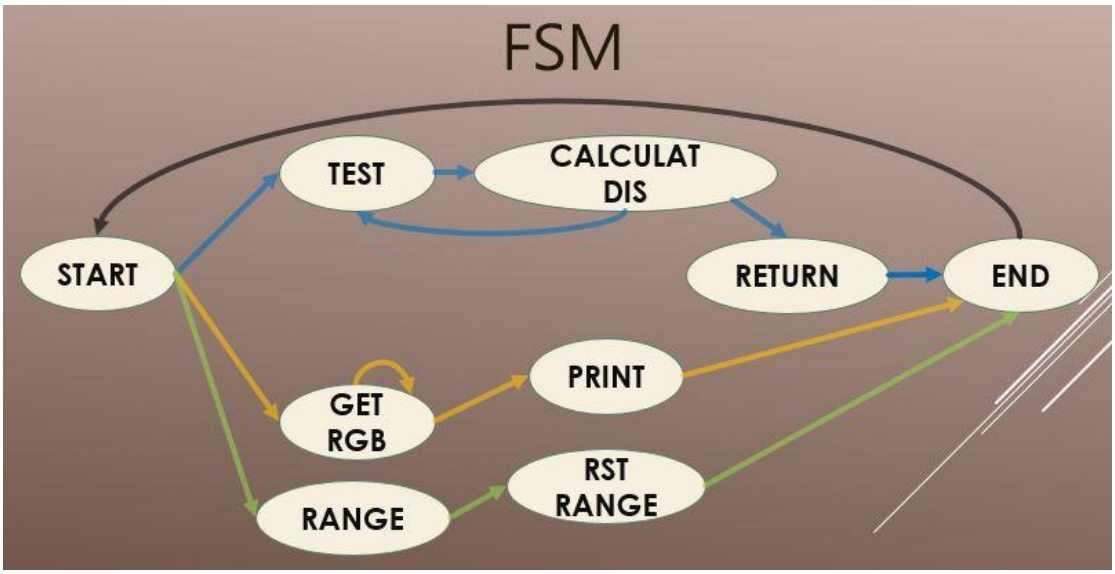
D. 從 node_dis[i][j]得到兩測資的距離，回到步驟 C 找下一個 child2。若是 Cw 的測資全部找到，回到步驟 B 找下一個 child1，接著再次計算該 child1 與所有 child2 的距離。

E. 找完 Cw 後，計算 Ck 與 Cw 的距離(參見上面所提的公式)，就換找下一個群聚。

七. Input、Output、Others (FSM)

1. Input :
   A. [31:0] pcpi_rs1, pcpi_rs2 : 輸入的 2 個數字。
   B. pcpi_insn_valid : 代表是否可以開始計算。
   C. [31:0] mem_rdata :
      藉由上一個 cycle 的 addr，從記憶體得到的值
2. Output :
   A. [31:0] pcpi_rd : 回傳值
   B. pcpi_wait : 若為 1，代表還沒算完。
   C. pcpi_wr : 若為 1，就將現在的 pcpi_rd 傳出去。
   D. pcpi_ready : 若為 1，代表已經算完。
   E. [31:0] mem_addr : 利用 addr 可以在下個 cycle 得到記憶體的值。
3. Others :
   A. [2:0] state, next_state
   B. [31:0] step, next_step
   C. [31:0] distance, next_distance
   D. [31:0] R, G, B : 該 pixel 的 RGB 值
   E. [31:0] MAX_R, MAX_G, MAX_B, MIN_R, MIN_G, MIN_B
      該群聚最大的 RGB 與最小的 RGB
   F. [31:0] next_MAX_R, next_MAX_G, next_MAX_B,
      next_MIN_R, next_MIN_G, next_MIN_B
   G. [31:0] return_value

八. FSM 中 state 的轉換



1. START ：

每次呼叫硬體 function 時，都會進入這個初始 state，pcpi_insn_valid=1，則下個 state 為 TEST，而且將 addr 設為編號為 pcpi_rs1 的圖片的記憶體位址，如此下一個 cycle 就能取得相對應的 pixel。

若輸入的 pcpi_rs2 是`print、`print_range 這類有被特別定義的數字，則分別進入 GET_RGB 或 RANGE。

2. TEST ：

將 mem_rdate 存入 test_pixel，下個 state 為 TRAIN，並將 addr 設為編號為 pcpi_rs2 的圖片的記憶體位址。

3. CALCULATE_DIS ：

將 next_distance 設為 distance+(text_pixel - mem_rdata)^2。

將 step+1 後存入 next_step，代表往該 pixel 的顏色(G or B)下繼續計算。

若 step==3 代表兩 pixel 已比完，則進入 RETURN。反之，則進入 TEST 繼續比對。

4. RETURN ：

將 return_value = distance 回傳回去，並把 next_distance 初始成 0。next_state 是 END。

5. GET_RGB ：

將 mem_rdata 存入 R、G、B(step = 1、2、3 時)，並在得到 RGB 時，和 MAX_R、G、B 及 MIN_R、G、B 時比較，以便得到該群聚的範圍。

若 step=3 代表已經取到 RGB 三色，next_state 為 PRINT。

6. PRINT ：

將存在 module 的 RGB 顯示出來，next_state 為 END。

7. RANGE ：

將存在 module 的 MAX_R、G、B、MIN_R、G、B 顯示出來，next_state 為 RST_RANGE。

8. RST_RANGE ：

把 MAX_R、G、B 初始成 0，MIN_R、G、B 初始成 255。next_state 為 END。

9. END ：

將 pcpi_wr=1、pcpi_ready=1 讓這次 module 結束。next_state 為 START。

九. 加速效果

1. 30 張

A. 軟

```
Cycle counter ........ 6039210
Instruction counter .. 1362482
CPI: 4.43
Status:DONE
```

B. 軟+硬

```
Cycle counter ........ 5116057
Instruction counter .. 1139334
CPI: 4.49
Status:DONE
```

2. 50 張

A. 軟

```
Cycle counter ........27881661
Instruction counter .. 6014548
CPI: 4.63
Status:DONE
```

B. 軟+硬

```
Cycle counter ........25618454
Instruction counter .. 5460171
CPI: 4.69
Status:DONE
```

可以看出加速效果大概是 10%左右。

十. 遇到的問題

十一. 討論

1. 如何確定結果是正確的

一種方法是自己也用手算，但如此太費時，因此每個群聚的 R、G、B 和三色總和範圍都顯示出來。結果如下:

A. 30 選 3

ROOT= 1

NODE= 1, R= 154, G= 126, B= 105

NODE= 4, R= 170, G= 168, B= 177

NODE= 5, R= 159, G= 150, B= 153

.

.

.

NODE= 11, R= 142, G= 172, B= 176

NODE= 29, R= 202, G= 202, B= 204

NODE= 30, R= 126, G= 122, B= 126

R: 86~ 202

G: 96~ 202

B: 99~ 204

ALL: 281~ 608

ROOT= 2

NODE= 2, R= 255, G= 253, B= 253

NODE= 16, R= 235, G= 235, B= 237

NODE= 21, R= 252, G= 249, B= 250

R: 235~ 255

G: 235~ 253

B: 237~ 253

ALL: 707~ 761

ROOT= 3

NODE= 3, R= 28, G= 37, B= 38

NODE= 7, R= 28, G= 30, B= 33

NODE= 10, R= 53, G= 54, B= 56

NODE= 13, R= 17, G= 17, B= 17

NODE= 19, R= 23, G= 47, B= 52

NODE= 24, R= 73, G= 71, B= 77

NODE= 27, R= 45, G= 42, B= 35

R: 17~ 73

G: 17~ 71

B: 17~ 77

ALL: 51~ 221

B. 30 選 4

ROOT= 1

NODE= 1, R= 154, G= 126, B= 105

NODE= 6, R= 164, G= 105, B= 118

NODE= 8, R= 134, G= 131, B= 128

.

.

.

NODE= 23, R= 126, G= 102, B= 117

NODE= 26, R= 131, G= 124, B= 116

NODE= 28, R= 128, G= 121, B= 138

NODE= 30, R= 126, G= 122, B= 126

R: 86~ 164

G: 96~ 142

B: 99~ 151

ALL: 281~ 457

ROOT= 2

NODE= 2, R= 255, G= 253, B= 253

NODE= 16, R= 235, G= 235, B= 237

NODE= 21, R= 252, G= 249, B= 250

R: 235~ 255

G: 235~ 253

B: 237~ 253

ALL: 707~ 761

ROOT= 3

NODE= 3, R= 28, G= 37, B= 38

NODE= 7, R= 28, G= 30, B= 33

NODE= 10, R= 53, G= 54, B= 56

NODE= 13, R= 17, G= 17, B= 17

NODE= 19, R= 23, G= 47, B= 52

NODE= 24, R= 73, G= 71, B= 77

NODE= 27, R= 45, G= 42, B= 35

R: 17~ 73

G: 17~ 71

B: 17~ 77

ALL: 51~ 221

```
 ROOT= 4
NODE=         4, R=       170, G=       168, B=       177
NODE=         5, R=       159, G=       150, B=       153
NODE=        11, R=       142, G=       172, B=       176
NODE=        12, R=       164, G=       162, B=       162
NODE=        18, R=       197, G=       198, B=       201
NODE=        20, R=       153, G=       174, B=       155
NODE=        25, R=       162, G=       164, B=       169
NODE=        29, R=       202, G=       202, B=       204
R:       142~       202
G:       150~       202
B:       153~       204
ALL:       445~       608
```

可以看出，每個群聚間的 R、G、B、ALL 幾乎都沒有重疊到，若是分的剩餘群聚數越多，每個群聚的範圍也會較小。

十二. 資料來源

1. http://mirlab.org/jang/books/dcpr/dcHierClustering.asp?title=3-2%20Hierarchical%20Clustering%20(%B6%A5%BCh%A6%A1%A4%C0%B8s%AAk)&language=chinese
   3-2 Hierarchical Clustering (階層式分群法)