電機二 郭笛萱
B03901009
0910368238
B03901009@ntu.edu.tw

資料結構與程式設計 期末專題報告

一、Design of simulation

Simulate 總共分為三個部分

1. 第一個部分為取得 pattern，當使用 file simulate 的時候，我會將 pattern 儲存在一個二維的 vector 中，當使用 random simulation 的時候，我設的 pattern 數的上限為 sqrt(NETLIST.size())，所以時間複雜度應為$\sqrt{n}$。
2. 第二個部分為將每個 gate 設值。我在每一個 gate 中存一個 unsigned long int 的_sim，專門儲存每一個 gate simulate 出來的值。從 NETLIST 的頭開始 simulate，將每個 gate 設值。
3. 第三個部份為找尋 fecGroups。首先，將所有 gate 丟進一個 fecGroup 中，然後去比較每個 gate 的_sim 一不一樣，再依據是否一樣去做分類。判斷是否一樣的方法是使用 hash，程式會去判斷 hash 裡是否已經有一個(Gate B)跟目前的 gate(Gate A)一樣的 key 的 data，如果有的話，就會把此 Gate A 塞進跟 Gate B 一樣的 group 中，如果沒有的話，就把 Gate A 丟進 hash，然後 Gate A 再另外新開一個 fecGroup。如何得知 GateA 在哪一個 group，並將 GateB 塞進跟 GateA 一樣的 group，我思考了陣子，最後決定在每個 gate 中存一個 groupNo.，去記錄每個組的編號，如此就可以將 GateB 丟進 GateA 所在的 group 中。
4. 最後當所有 gate 都被 search 一遍後，程式會去 collectFecPair，將 size >= 1 的 pair 丟進一個新的 vector 中，原來的 vector fecGroup clear 之後，再把這個新的 vector 中的內容丟回 fecGroup。會使用這個方法而不是 erase 的原因為，erase 十分的耗時間，每次呼叫一次 erase，都要重新把後面的資料往前移，在把 erase 去掉後，速度快了三倍左右。

二、Design of fraig

1. Fraig 加速構想：Fraig 我原本有做加速設計，但是後來因為發現測資出來的結果有問題，所以放棄了加速的版本。加速的構想為：一開始，將 fecGrp 按照 netList 的順序 sort 了一遍，利用每個 fecgrp 裡面的元素與 PI 的距離，決定要怎麼 sort，與 PI 距離最小的 fecGrp 先開始 proof SAT，如此從頭開始找應有助於電路的簡化。再來，每做

$\sqrt{NETLIST.size()}$的 fecGrp，被證明出相同的 pair，merge 一次，並且重新收集造成 SAT 的 input pattern，然後再 resimulate 一次，如此一來，可以減少 fecPair 的數量，使速度加快。

2. 實際上的設計為：從 fecGrp 的一開始證明兩個 gate 是否相等，當證明是相等後，就把兩個 gate merge 起來。若兩個 gate 不相等，我會去計算這個 group 不相等的次數，超過一定次數就放棄此 FecGrp，直接把這個 group pop 掉，跳到下一個 fecGrp 繼續做。

三、Experiment

1. 實驗一：simulation 速度
   (1) 實驗設計：探討 sim13 和 sim12，gate 數量與 simulation 所需的時間之間的關係，並與 ref program 比較
   (2) 實驗預期：預期時間複雜度為 O(n)，因為當 pattern 讀進來後，為每個 gate 設值，應為 O(n)，然後在分 fecGrp 時，每個 gate 只須看一次即可分完，因為 simulation 的時間複雜度為 O(n)。
   (3) 實驗結果：

   sim13.aag 的 simulation 速度 (gate size : 88410)

My program :	Ref program :
Time used : 26.56s	Time used : 12.9s
Memory used 198.8M	Memory used : 10.42M

   從整體時間來看，速度大約是 ref program 的兩倍，如果不去處理 fecGroup 的話，時間大約是 10s，所以時間大部分是花在處理 fecGroup 的部份上。

   sim12.aag 的 simulation 速度 (gate size : 9642)

My program :	Ref program :
Time used : 2.46s	Time used : 1.99s
Memory used 17.44M	Memory used : 2.137M

   從整體時間來看，速度大致跟 ref program 相同，如果不去處理 fecGroup 的話，時間是 0.51s，所以同樣的，有超過一半的時間是花在分類 fecGroup 上。
   若比較這 sim13, sim12 這兩筆測資，可以發現兩者的 gate 數量差十倍，而時間也大約差十倍，所以 simulation 的速度與 gate 數量呈線性關係，時間複雜度為 O(n)，結果符合實驗預期。

2. 實驗二：fraig 速度
   (1) 實驗設計：探討 sim12 與 sim09，fecGrp 的 size 與 fraig 的速度間的關係，並與 ref program 比較。

(2) 實驗預期：預期時間複雜度為 $O(n^2)$，因為我是採取每個 fecGrp 中的每一個 fec pair 都要證明一次的方法，所以應跟 n^2 成正比。

(3) 實驗結果：

Sim12.aag 的 fraig 速度 (gate size : 9642, fecGrp size = 2594)

My program :	Ref program :
Time used : 213.1s	Time used : 13.49s
Memory used : 17.25M	Memory used : 5.281M

Sim09.aag 的 fraig 速度 (gate size : 3587, fecGrp size = 338)

My program :	Ref program :
Time used : 3.17s	Time used : 0.78s
Memory used : 17.25M	Memory used : 5.797M

當我在寫 fraig 時，因為沒有使用優化的功能，所以時間會花的非常多，使用 sim12 與 sim09 比較可發現當資料量變大時，我的程式所花的時間約為 n^2 倍，但 ref program 所花的時間約為 2n 倍，時間相差非常多。

四、Bottleneck

1. 寫程式的過程，碰到的最大的問題應該是如何加快 simulation 的速度。一開始寫完 simulatioon 的時候，速度十分的慢，大概是 ref 的五倍左右，所以我就去思考哪裡寫得不好，後來將 erase 改進後，速度果然快了許多。
2. 資料儲存及管理是一個很大的困難點，在寫 hw6 的時候，因為不需要將 gate 的數量更動，所以我用了很多個 vector 去儲存不同種類的 gate 型態，讓在寫 hw6 的時候可以快速取得資料，但是在做 final project 的時候，因為 gate 的數量需要更動，所以時常會漏掉更新那些 vector，造成 segmantation fault 的情形，所以這讓我學習到不是存越多東西越好，這會讓程式的結構十分凌亂。