FRAIG REPORT

學號：B02901013

姓名：鄧傑方

手機：0955-387-923

信箱：b02901013@ntu.edu.tw

1. 概述

與Homework 6相比，FRAIG主要增加了五項新功能：CIRSWEEP, CIROPTMIZE, CIRSTRASH, CIRSIMULATE, CIRFRAIG，感覺功能強大了許多，也複雜了許多！

在最初parse檔案進來時，我利用map將GateID與CirGate\*連結在一起，而最後進行了一次DFS，紀錄走過的Gate。每次進行前，都先setglobalref，將globalref+1，就可以利用globalref來判斷是否被走過，建造出了一個NetList的vector來存整個DFS的順序。

1. CIRSWEEP

第一個部分先做CIRSWEEP，每次sweep時先做一次DFS，就可以判斷哪些Gate是沒走到的，將這些Gate的fanin、fanout進行與其他Gate的連結處理後就可以從map中移除，完成sweep的部分。

三、CIROPTIMIZE

第二部分的optimize，我先設了一個bool doOpt，決定在做DFS的時候要不要進行optimize的動作，若要進行optimize則判斷此Gate屬於四種狀況(fanin是否為相同Gate、是否有接const…等)，若有符合，則將其fanin、fanout進行處理、連結，相較於CIRSWEEP處理fanin、fanout會複雜一些，最後將optimize產生的多餘Gate從map中移除，完成此一Gate的optimize。當走完DFS即可完成此電路的CIROPTIMIZE。

四、CIRSTRASH

第三部分的strash，須由自己設計出一個key，來對應到\_buckets中的位置，我使用的方法是將兩個fanin的GateID相乘，如果有not閘，則將GateID+1後再相乘，\_numBuckets則是AIG數量的兩倍，實際使用起來在Gate數量不多時比較容易分在同一個\_buckets中。與CIROPTIMIZE類似，我設了一個bool doStr，決定在做DFS的時候要不要進行strash，產生HashNode，並存入\_buckets中，再存入之前，我會先判斷在\_bucket中是否已存有相同的key，若有且其fanin也確實相同，則用\_buckets中存在的Gate將此Gate進行merge的動作，若否則將此HashNode存入\_buckets中。

五、CIRSIMULATE

第四部份的simulate真的複雜了好多，一開始我先在CirGate新增了size\_t\* value的member，並以二進制的形式來存-file或-random產生的模擬資料，每32個pattern為一組，這樣就可以直接使用&來加快模擬的速度，並將PI的value處理好之後，才真正進入電路的simulate，產生FECgroups。

PI有了模擬資料後，我一樣是設了一個bool doSim先進行DFS，將每個Gate的value模擬出來，並使用第一組測資(前32個pattern)模擬的結果，也就是每個Gate中的value設計成key，存入Hash中，方便我直接找到FEC pairs，因為要考慮IFEC的問題，所以每個Gate我都將其key,~key存入Hash中，使用上比較直覺，可是會有一些要處理的問題，在速度上也會慢一些。

接著用vector< vector < pair <HashKey , CirGate\* > > >作為FECgroups，若在\_buckets中的FEC pairs小於兩組，就可以直接砍掉，最後就可以得到用第一組測資(前32個pattern)模擬出來的FECgroups。

之後就繼續進行其他組測資的模擬，一樣是每32pattern為一組來模擬。此時就只要再針對FECgroups中的每個FECgroup進行，方法類似做第一組測資的方式，將此組測資產生的value一樣作為key，存入Hash中，並判斷是否可以將此FECgroup分成更小的FECgroup，之後將單獨FECpair砍掉，這樣就大致完成最終所有測資模擬後產生的FECgroups。

第一組測資

PI處理

每個Gate中的value

將value建成key，存入Hash

利用新測資產生的value

針對每個FECgroup做Hash建成更小的FECgroup

得到第一組測資產生的FECgroups

其餘測資

所有測資模擬後產生的FECgroups

接著就能進行sat

比較值得注意的是模擬pattern的數量吧，因為在測試sim13.aag時，光是跑4xxx個pattern就要花上15秒左右，跟老師的20000出頭pattern花4~5秒相比慢了許多，所以我pattern的數量並不是由存在的FECgroups數量來決定，而是由PI的數量經過數學運算來取得一個合適的pattern數量，不過當然pattern跑得少，最後cirp –fec的數量就會比較多，跑出來的結果我的有44xx左右的FECgroups，而ref則只有38xx左右！

Cirgate&Cirp –fec算是經過simulate之後需要特別去處理的功能吧！Cirgate印出value值時，我是把最後模擬的那組value印出來，若是-file的pattern倍數並不為32時，在PI的時候則會補上0。而因為Cirp –fec有要求要按照GateID順序排列，所以每次insert到Hash中時，除了找到對應的\_buckets，我還由後往前比GateID的大小決定其位置，之後用vector < pair <HashKey , CirGate\* > >裝FEC pairs時比較方便。

六、CIRFRAIG

之前一直想說simulate做完之後，fraig應該很快，真正做的時候才發現不是這樣，光是研究sat就花了不少時間，之後就可以利用sat幫我們證明FECgroup內的Gate是否一樣。基本步驟就是對每一個FECgroup，將其中的Gate兩兩抓出來做sat比較。因此就每一個FECgroup我先用他的第一個Gate來與其他Gate比較，若是UNSAT則代表這兩個Gate完全一樣，先繼續留在此FECgroup中，若是SAT則將Gate移除，並存到新的一個FECgroup(tmp)中，最後存在此FECgroup中的就會只剩完全一樣的Gate了，而若是tmp中的Gate大於一個，則將此tmp push\_back到FECgroups中，最後FECgroups中的每個FECgroup就只會存在完全一樣的Gate，接著就可以進行merge的動作。

不過我卻在這裡崩潰了，一直crash…，紀錄一下現在時間2015.1.20 04:03，先去睡個覺好了，大概一兩年沒這麼晚睡了…，希望早上醒來可以把它完成。

現在時間2015.1.20 11:55，大致上把之後merge的動作完成了。在得出FECgroups中的每個FECgroup都只存在完全一樣的Gate之後，利用先前得到的DFS時AndGate GateID的順序，進行merge的動作，每走一個AndGate就判斷要用哪個Gate進行merge，用的是在此FECGroup中DFS最早走到的那個Gate，我想之前會crash可能就是沒用DFS最早走到的Gate來merge，然後亂連接，導致電路爛掉。這個方法感覺有很多缺點，因為每次都要在FECGroup中找到DFS最早走到的Gate，再加上FEC、IFEC皆會存在FECGroups中，使sat證明的數量變成兩倍，也沒有記錄重要的pattern，不過死線在即，只能先求有部分功能，之後來得及的話再來慢慢修！

後來又改了一些地方，改成先不證明，而是利用先前得到的DFS時AndGate GateID的順序，每走一個AndGate也是會進行判斷，若此Gate為其FECgroup中進行DFS最早走到的那個Gate時，則在此時進行這組FECgroup證明的動作，即可少掉大概一半的證明數量，不過做完剩餘的AIG數量總是跟ref不一樣，真是奇怪！

七、performance比較

(1) CIRSWEEP

Cirsweep感覺起來跟ref的差不多，不過因為我要再跑一次DFS來判斷哪些Gate是走不到的，所以在sim13時，雖然沒remove任何Gate，可是也會花上0.81秒

DFS的時間。

(2)CIROPTMIZE

感覺也是跟ref差不多，應該差在DFS的時間。

(3)CIRSTRASH

測sim13.aag時雖然沒有任何可以strash的Gate，但還是花了1.5秒，主要可能還是DFS的時間。

(4) CIRSIMULATE

因為我的設計方式是每模擬完一筆測資(32pattern)就會存入Hash一次，然後再得到比較小的FECgroup，可能就是在重複進行這個動作時把時間都耗掉了，用usage查看的結果，記憶體使用量也暴增了不少，有時間的話應該這個地方有很多可以改進的地方，目前有個想法，若是連續跑個10筆左右的測資，若FECgroups的數量一直沒有減少，應該就能推測極難找到可以分辨他們的pattern，此時就可以進行fraig以減少時間。測sim13的結果就如同上面我所述。

(5) CIRFRAIG

一開始用DFS幫SatSolver建電路。接著對每一個FECgroup之中的第一個Gate與其他Gate進行證明，時間複雜度O(N)。UNSAT的Gate就繼續放在此FECgroup中，最後才會merge；SAT的話，就會移除並存入新的一個FECgroup(tmp)。整個FECgroup比對完，tmp的size有大於1，就將tmp push\_back到FECgroups。

跑比較小的電路都能證出來，在測sim13時則直接卡住了，有太多可以加速的小技巧都沒用，也難怪會卡住，有時間一定要把它修到好！

改成新的方式後，又測了一次sim13，可以看的到他在緩慢進行merge了！

八、心得

上完了這一學期的資結，我想增進的不只是程式的能力，更重要的還有查資料、自我學習的能力，尤其是一開始的幾個homework，很多東西都是大一計程沒有用過的，像是之前都沒用過vector、map……等，這個時候就要一直爬文，研究如何使用，過程真的學到很多。此外就是如何妥善安排時間，以及那種真的花20小時up的時間完成一件homework的決心及毅力吧，這真的很磨練人的耐心，對於電機系的學生也很重要，以後想必會有很多project可能要花更長時間，尤其我覺得我沒什麼天分，前幾個homework都是將時間砸下去，努力完成所有功能，將遇到的bug處理掉，所以分數都能拿到八成以上，過程真的很艱辛，不過看到最後寫出來的作業在功能上幾乎都能與老師相同時的那種喜悅感(速度另當別論)，真的都快喜極而泣了，也很有成就感。

FRAIG果然不是徒具虛名，在過年的那幾個連假我就寫完了CIRSWEEP、CIROPTIMIZE、CIRSTRASH，想說利用考完的五晚上、六日一二整天將整個FRAIG完成，豈知CIRSIMULATE的功能一下子複雜了好多，在處理效能上也花了很多時間，而CIRFRAIG更是崩潰了好久，只能完成一部份的功能，在效能的處理上還是無法改進，比較大的電路都證不出來，最後沒能將CIRFRAIG做到完整，還滿遺憾的，感覺就是有種缺陷，還有很多進化的空間。

很高興在大二就修了資結這們課，比較能感受到身處電機系的那種感覺了，很累但是我很喜歡，每次作業都能感受到自身能力的進化，真的很謝謝老師，超級扎實的，好課大推XD！