DSnP Final Report

舒泓諭

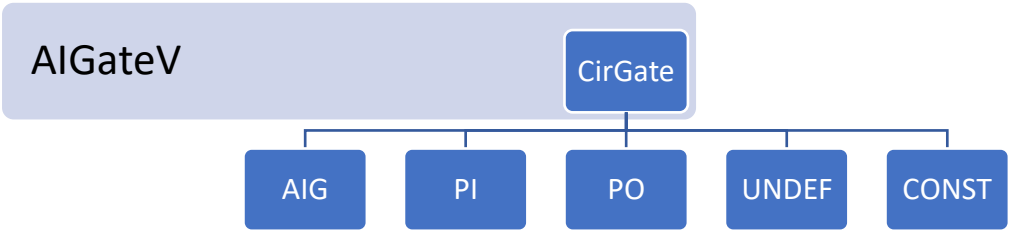
BO5602052

0983665308

整體設計方向：

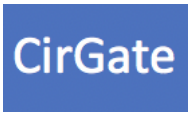
本次因為眾多事情撞在一起，所以開始寫 **fraig** 的時候就決定以，在最短的時間能完成 **fraig** 功能的實踐，而不太追求效能，尤其是 **fraig** 的 **memory overhead** 和 **runtime** 都是教授的數倍以上。

資料結構：

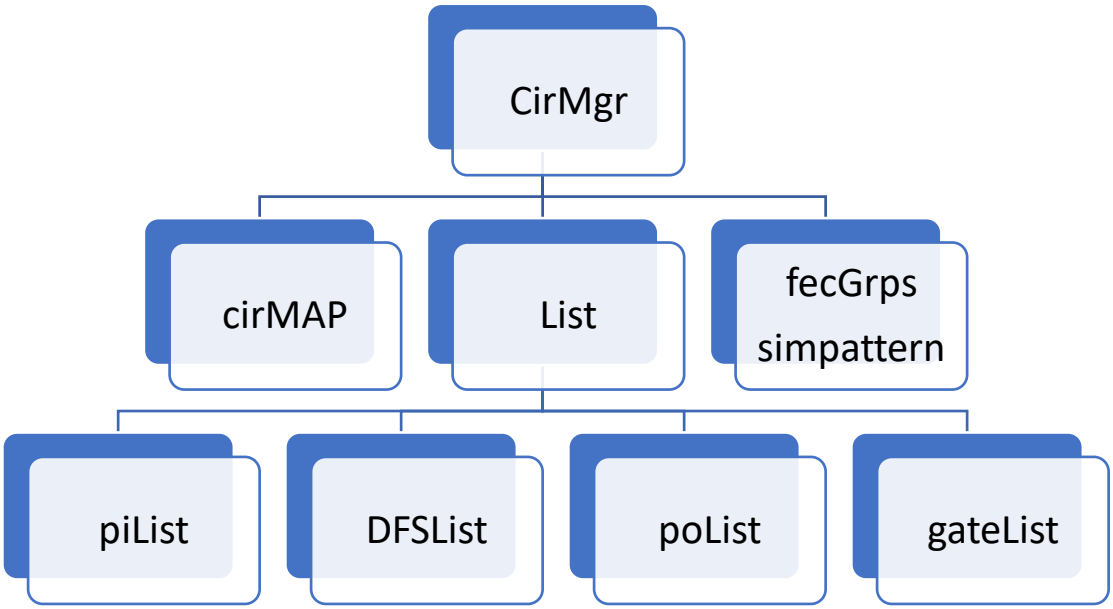


總共有兩種主要的 **data type**：分別是 **AIGateV**、**CirGate**

AIG、**PI**、**PO**、**UNDEF**、**CONST** 分別繼承、**CirGate**，我幾乎所有的 **data member** 都統一定義在 **CirGate**，雖然這樣寫起來 **code** 十分的簡潔易懂，但到後面 **sim13** 的 **fraig** 時記憶體用量真的十分的可觀。



在 **CirGate** 之中，**fanin**、**fanout** 分別用一個 **vector** 來存他，雖然 **fanin** 基本上在本次作業中一定只會有兩個，但當初在寫的時候在想說，這樣程式的可讀性比較高而且我比較喜歡用 **vector** 來存東西，所以糾結了老半天還是用了 **vector** 來寫。此外，**fanin**、**fanout** 所存的 **data member** 其實是不一樣的，**fanin** 存的是 **AIGateV**，這邊會這樣存有很大一部分因素是因為，考慮 **gate inverse** 的問題，存法就是用教授投影片上的，用 **pointer** 的最後一個 **bit** 來存。



在 **CirMgr** 裡面我總共存了三種類型的資料：

（1）List:將 gate 分成不同的類別依照屬性存入不同的 vector，但每次再 Ciroptimize 完後都會再更新清單裡面的東西，雖然維護這些清單很麻煩，但用起來真的蠻方便的。

（2）cirMAP:會將 gate 的 ID 對應他的 pointer 存入以 STL 裡面的 unordered map 存入，因為這種 map 其實是一種 hash table，所以隨機存取資料是 O(1)。

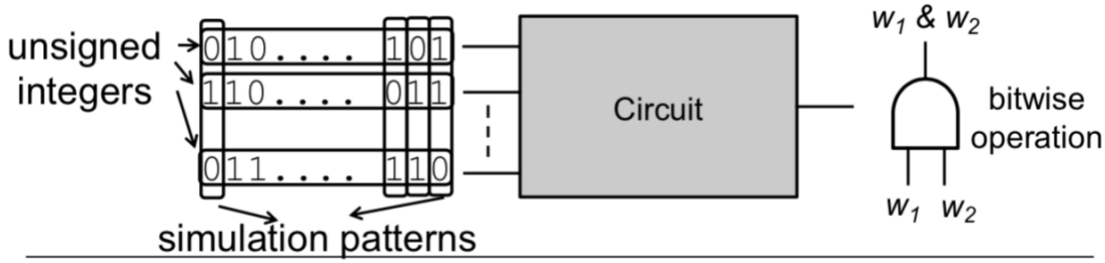
（3）fecGrps & simpattern：這兩個也是 vector 會在 cirsimulate 的時候更新 fecGrps 內的資料，並在做 cirfraig 的時候，把證明完是 UNSAT 的稀有 pattern 搜集在 simpattern 裡面。

（4）其餘還有存一些資料，因為太瑣碎就不放上來了。

演算法：

Simulation:

在這邊模擬所需要用的 **pattern** 會是裝在 **size_t(unsigned integers)**，再利用 **Hashkey** 這一個資料結構，去轉換成對應的 **FEC pair**。



CirMgr 裡面存的_fecGrps，是用 vector<vector<CirGate＊>>，來存不同的 fecgroup，雖然 simulation 有兩種模式：（1）random mode、（2）pattern mode，但大體來看都是由下面三個步驟組合而成的。

(1)初始化_fecGrps：

模擬第一組 pattern，對於每一個在_DFSlist 的 gate 去進行 simulate()，依照不同的 pattern，利用 hash 分成不同的類別。最後每一組有超過兩個以上的 FEC pair，存入_fecGrps，其他只有一個的 pattern 直接把他砍掉。

(2)模擬 _fecGrps：

先創造一個_newGrps，接著對_fecGrps 的每一個 FECpair 去做去進行 simulate()，每一組有超過兩個以上的 FEC pair，存入_newGrps。最後把 _newGrps 換到_fecGrps。

(3)排列 _fecGrps：

最後將_fecGrps 裡面的 gate 按照 ID 大小排列。

Fraig:(SAT Engine)

利用 SAT Engine，可以藉此證明在 FEC pair 的兩個 gate 是不是一樣的。如果經過證明兩個 gate 被證明是 UNSAT 那麼，則兩個 gate 可以被 merge 在一起（這邊的 merge 方法和 strash 的方法是一樣的），然而我每找到一個就 merge 一個，會導致最後的 runtime 非常的可觀，例如：sim13 就跑了 2850sec。如果經過證明兩個 gate 被證明是 SAT 的話那麼這一組 FEC group 是可以被拆開的，並把這一些 pattern 存在_simpattern 裡面。

實驗：

測試 simulatio＋fraig 時間會和電路大小成正相關，驗證其時間複雜度是 O(n^2)

電路(gate 數)	時間(sec)
Sim15（886）	4.83
Sim06(4270)	401
Sim12(9364)	2850
Sim13(81710)	來不及跑完....