FPGA Final Project report

111062615 黃奕翔

**Introduction**:

這次的作業主要是要我們將circuit nodes放入Fpga中,並且在不違反三個規定的情況下,希望讓Fpga的sum of external degree最小。

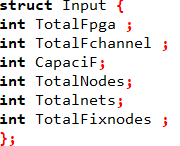
三個規定如下:

1. Fixed node constraints: There are some circuit nodes which are already assigned to specific FPGAs and cannot be moved to other FPGAs.

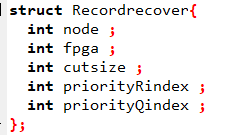
2. Partition size constraints: The total number of circuit nodes assigned to each FPGA must not exceed the capacity of a FPGA (we assume all FPGAs have the same capacity).

3. Topology constraints: If node u is the source node of a net and node v is a sink node of the same net, then nodes u and v are either assigned to the same FPGA (i.e., FPGA(u) = FPGA(v)) or assigned to two neighboring FPGAs (i.e., (FPGA(u), FPGA(v)) ∈ E’)

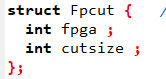
**Data Structure:**



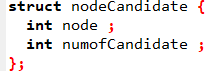
用來存6個input資訊。



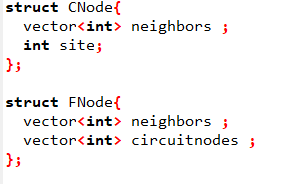
用來記錄之後可能需要被recover的資訊



用來記錄某個node放入某個fpga後其cutsize的變化量



用來記錄某個node的Candidate數



用在剛開始建圖時的哪些node應該和誰相鄰以及哪些FPGA應該和誰相鄰



用來存放每個FPGA還剩下多少的Capacity



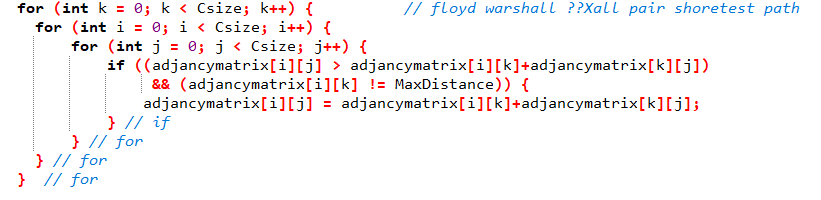
用來存放node可以放入的FPGA(放入後不會違反第三個Topology constraint的情況)

**Process:**



這個Constructor是用來建circuit node的Graph以及FPGA的Graph,裡面會將每個circuit的neighbors以及FPGA的neighbors設定好,以便後面的Function 利用。

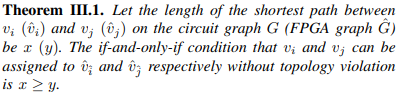


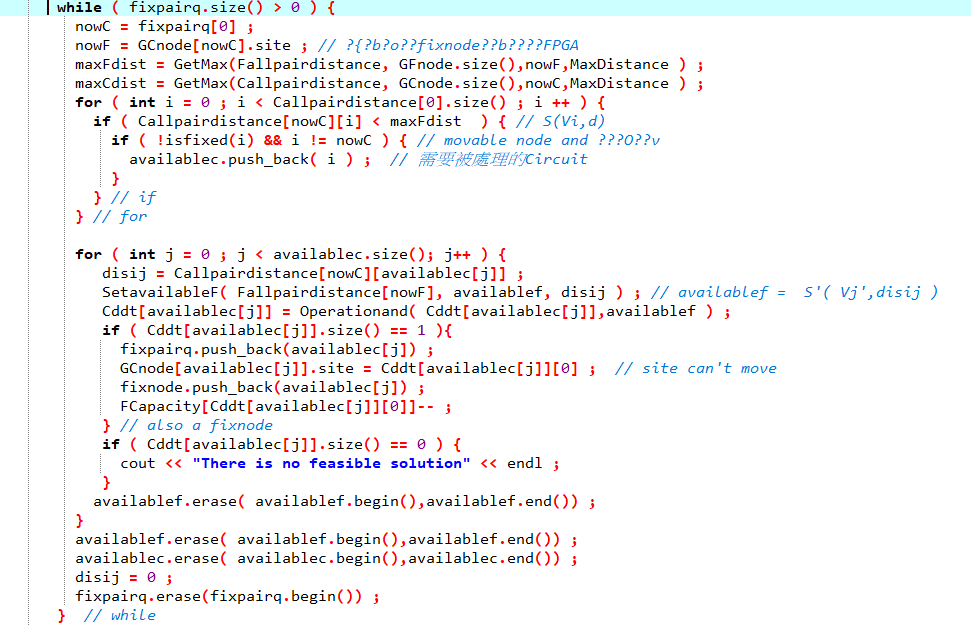


這兩個Function是用來找出每個circuit node之間的距離以及每個FPGA之間的距離,在這邊我是利用Floyd warshall的演算法去找出距離的。



這個Function則是用來設定moveable節點在Fixed node已經放入他們各自的Fpga後,在不違反Topology constraint的情況下可以放在哪些Fpga,而這個Function最重要的就是不違反Topology constraint,要做到這件事必須滿足Theorem III





首先就是先找某個fixed node放置的FPGA : Fi與其最遠的FPGA : Fj的距離maxFdist,接下來再找出所有這個fixed node與其相鄰不超過maxFdist的所有點,將他們放入到availablec中表示這些點需要被更新他們可以放置的Fpga,才不會違反Topology constraint,下面的code在做的就是利用上面Theorem III的定理去將這些點的Candidate FPGA更新,直到所有fixed node的點都檢查完以及和他有關的moveable node都更新完,則這個Function結束。



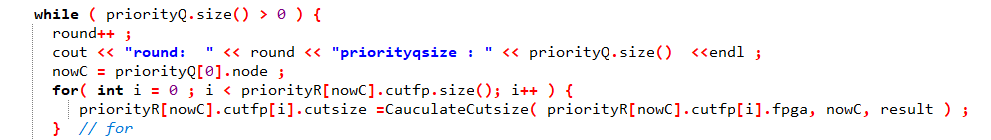
這個Function是最後用來放置circuit node的，首先會先定義兩個Priority Queue 分別是

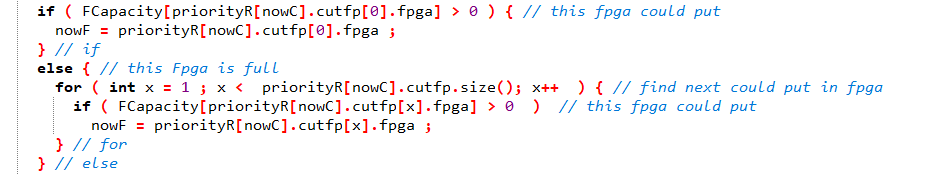


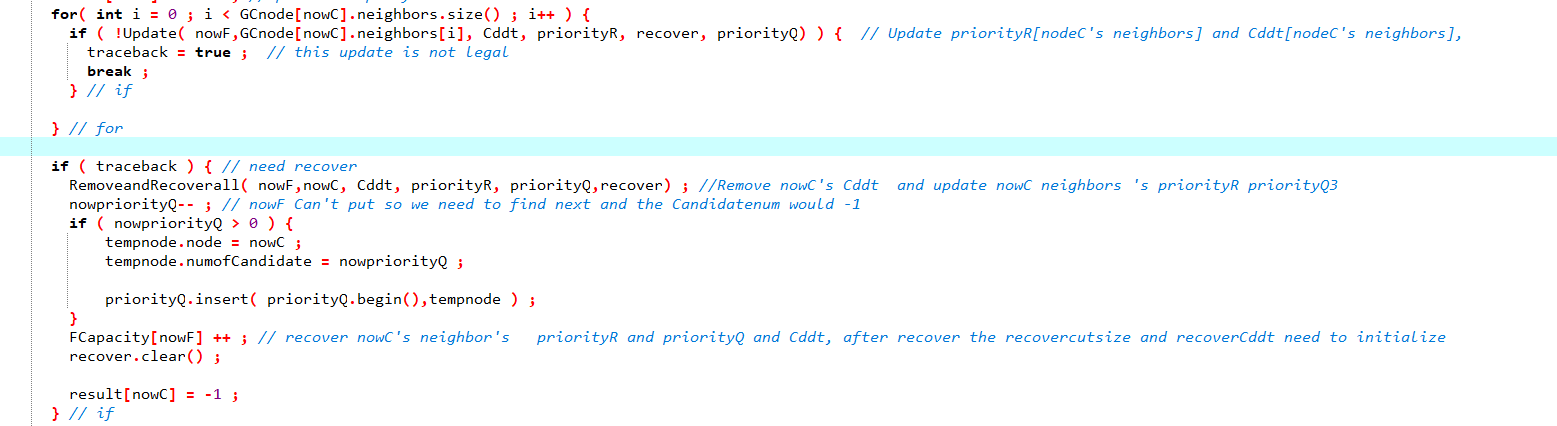


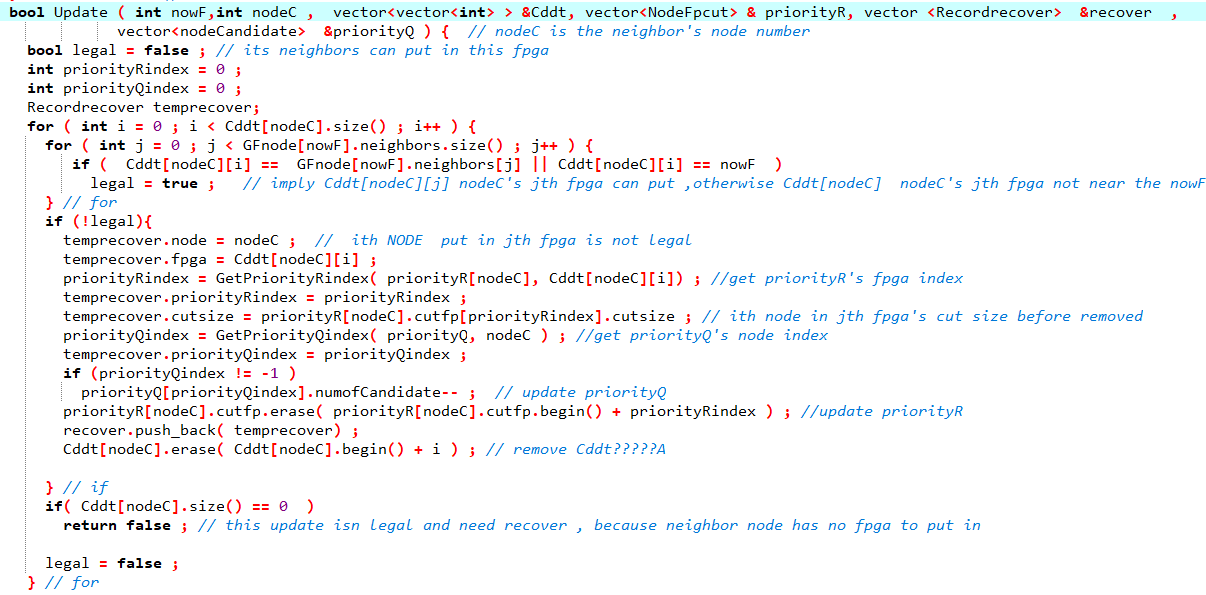
PriorityR : 是用來存放某個circuit node放入某個Fpga後其cut size會增加多少,然後根據cut size增加的量來給予Priority,增加的越少其priority越高,增加的越多其priority越低。

PriorityQ : 是用來存放circuit node,若node的Candidate數目越少則其Priority越高,表示必須越早放入FPGA否則可能違反Topology constraint.

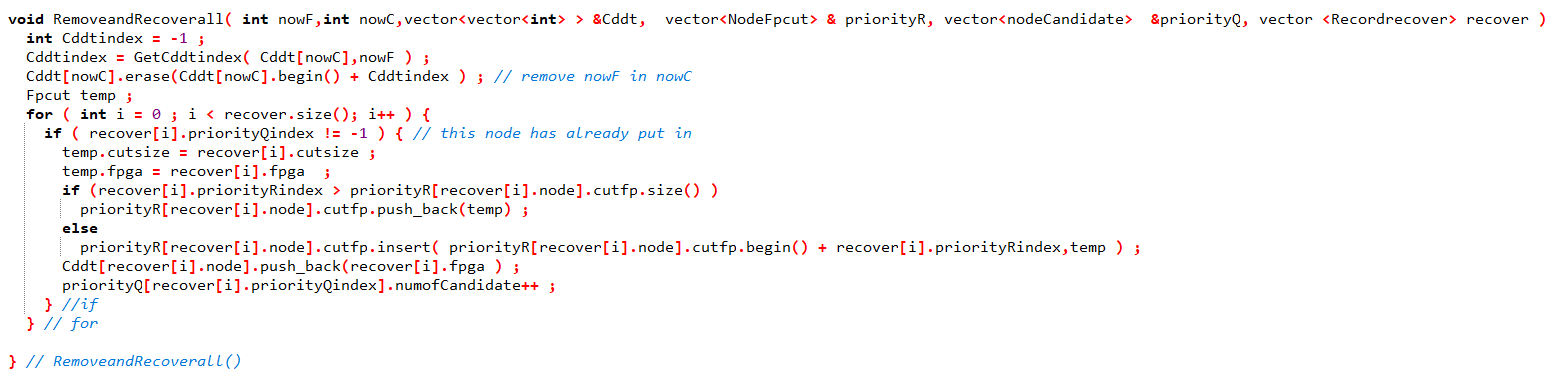


接下來這個Function的結束就是盡可能地將每個node都放入FPGA,也就是直到PriorityQ的size為0則結束,每一round會取出一個node,並將node放入FPGA,首先要確認這個FPGA是否為FULL,若是的話則找次高Priority的FPGA。且若放入某些Fpga其 cut size增加的量一樣,則會選擇被最少node當作Candidate的那個Fpga,因為這樣就不容易搶資源。





確認放入後就要去更新這個circuit node的neighbors,將他們的Candidate做更新(避免違反Topology constraint),以及將一些recover的資訊存起來(recover的資訊是用來避免萬一現在這個node放入這個FPGA後他的neighbors沒有FPGA可以放的時候就需要將其recover回尚未放入後的樣子),也將PriorityR以及PriorityQ去做更新。

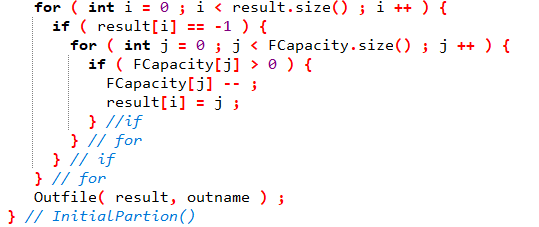


若需要recover則會進入上面這個Function,這個Function主要做兩件事:

第一個:就是把這一round放入的某個fpga從這一round的node 的Candidates中刪除,表示他未來不可以再放入這個FPGA。

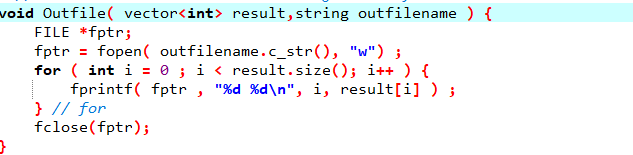
第二個:將他的neighbors還原成原本的樣子以及將PriorityQ以及PriorityR也一併還原。

一直重複以上的事直到所有Node都盡可地放入他們可以放入的FPGA後,則迴圈結束。



迴圈結束後倘若有Node真的無法被放入任何一個FPGA則會用隨機放的機制,這個機制是在不違反前兩個限制的大前提下去做Placement。

以上大概就是整個放置circuit node的流程。



放完後會再將result寫入Output檔案中,即結束整個程式。