VLSI HW3

1. 吳律穎 111062697

(2)

--HOW to Compile

In HW3/src/ , enter the following command:

$ make

It will generate the executable file "hw3" in "../bin/".

If you want to remove it, please enter the following command:

$ make clean

--How to execute

In this directory, enter the following command:

Usage: ../bin[exe] [hardblocks] [nets] [pl] [floorplan] [dead\_space\_ratio]

e.g.

/bin/hw3 ../testcases/n100.hardblocks ../testcases/n100.nets ../testcases/n100.pl ../output/n100.floorplan 0.15

In "HW3/bin/", enter the following command:

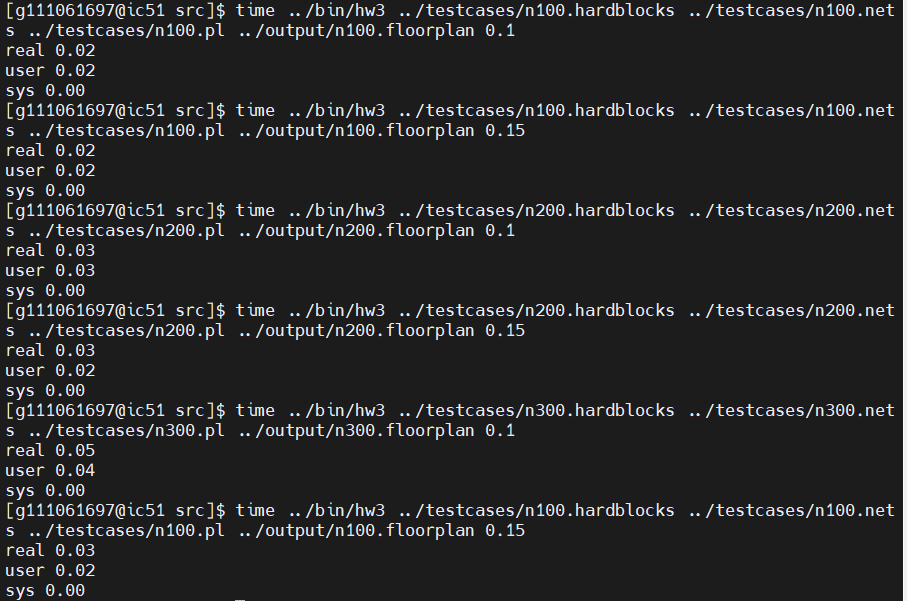
Usage: ./[exe] [hardblocks] [nets] [pl] [floorplan] [dead\_space\_ratio]

e.g.

./hw3 ../testcases/n100.hardblocks ../testcases/n100.nets ../testcases/n100.pl ../output/n100.floorplan 0.15

(3)

如下表所示，不論是建立結構或是初始化polish所花的時間都極小趨近0%，主要的時間花費都在透過polish計算floorplan面積長度和cost的時間，I/O時間也很少幾乎沒有，要讀到較大的檔如n300才有較多一些的時間



(4)

n100\_0.1 block\_area : 179501 Minarea: 193158 ratio : 0.076

n100\_0.15 block\_area : 179501 Minarea: 198414 ratio : 0.105

n200\_0.1 block\_area : 175696 Minarea: 191406 ratio : 0.089

n200\_0.15 block\_area : 175696 Minarea: 193984 ratio : 0.104

n300\_0.1 block\_area : 273170 Minarea: 294833 ratio : 0.079

n300\_0.15 block\_area : 273170 Minarea: 302978 ratio : 0.109

最小為 0.076

(5)

我初始化的polish與他不同，我是盡量將同高的block擺一起，碰到outline邊緣就往上繼續疊，是為了在開始就提供一個接近方形且面積小的

solution。

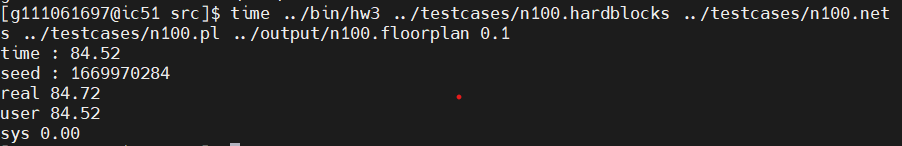
我的cost function與他不同，在超出outline的情況下area和 wire的cost和最大只能為1，超出outline的penalty為max(超出的高 , 超出的寬)^2，

而若在outline中penalty為0，這種penalty可以讓形狀維持在接近正方形，並且會先收縮到outline中再優化area或wire。

(6)

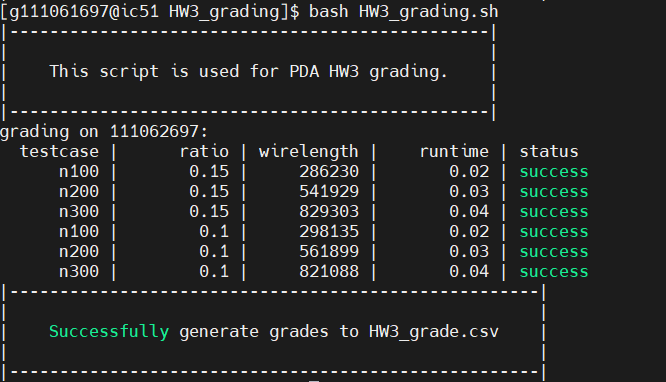
如(5)所述，照這種初始化的排法，原始floorplan就已經為一個在outline中或接近outline的解，所以可以大大縮短時間。

若不使用結果如下



Wirelength為309559

(7)

我的結果： 

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Wirelength | | | Runtime | | |
| n100 | n200 | n300 | n100 | n200 | n300 |
| 286230 | 541929 | 829303 | 0.02 | 0.03 | 0.04 |

與前三名比較 (增加或減少的 %)：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Wirelength | | | Runtime | | |
| Ranks | n100 | n200 | n300 | n100 | n200 | n300 |
| 1 | + 38% | + 49.3% | + 62.2% | - 99.85% | -99.96% | -99.98% |
| 2 | + 36.7% | + 42.7% | + 58.9% | - 99.92% | -99.97% | -99.98% |
| 3 | + 36.1% | + 38.1% | + 52.1% | - 99.94% | -99.98% | -99.99% |

我的結果在長度方面輸給了它們，但於時間方面有優勢，可能是由於我的跳脫條件為進到outline就結束，但在進outline前我的cost function不會對wire 或 area直接給予好處或懲罰所導致

(8)

透過不斷的嘗試，學到了不同cost function對於結果的影響會有多大，也知道initial的差別會對於速度和quality有非常直接的關係，也學到不同temperature變動的策略。對於上述的各個因素，都進行了多次的測試，最後才能達到規定的constraint。