(1) 王竑智 109062542

(2) type “make” to compile

in /HW3/src, type “../bin/hw3 *.hardblocks *.nets *.pl *.floorplan dead_space_ratio” to execute

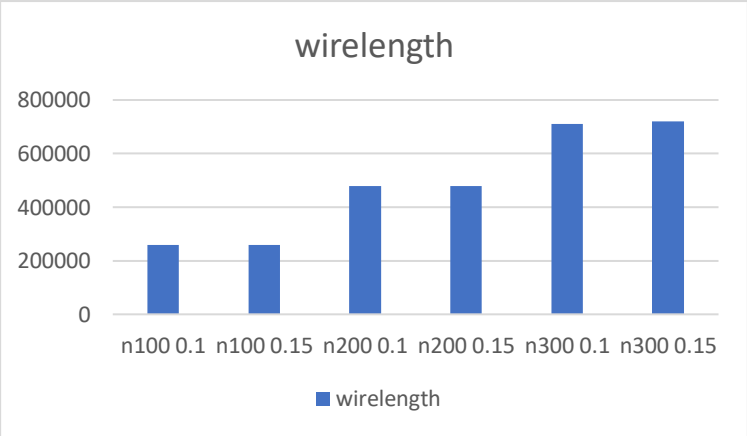
e.g ../bin/hw3 n100.hardblocks n100.nets n100.pl n100.floorplan 0.1

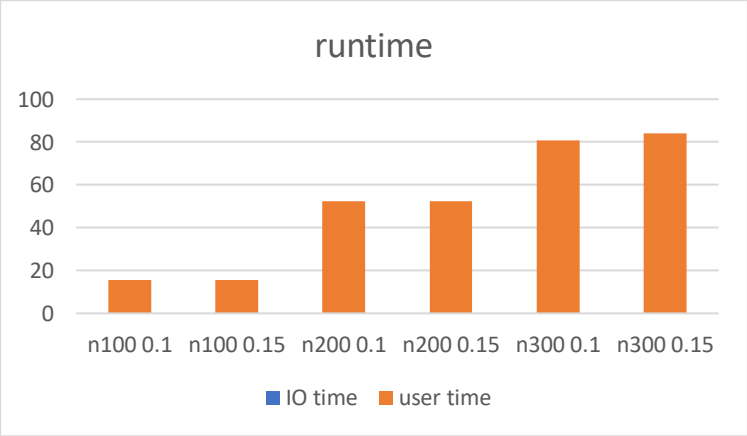
in /HW3/bin, type “./hw3 *.hardblocks *.nets *.pl *.floorplan dead_space_ratio” to execute

e.g ./hw3 n100.hardblocks n100.nets n100.pl n100.floorplan 0.1

(3)

case	dead space ratio	wirelength	runtime	IO time	user time
n100	0.1	258563	15.55	0.02	15.53
	0.15	258407	15.71	0.02	15.69
n200	0.1	479455	52.23	0.01	52.22
	0.15	478639	52.45	0.01	52.44
n300	0.1	711540	80.76	0.02	80.74
	0.15	720591	84.08	0.01	84.07

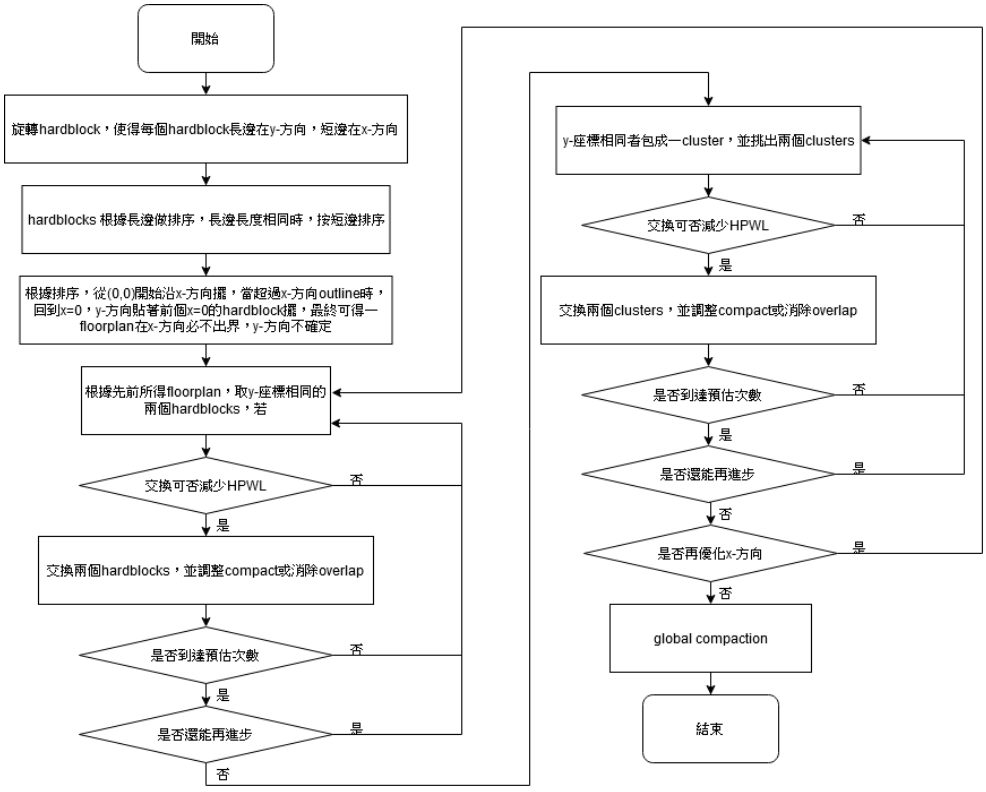




從 runtime 的長條圖可看出 IO 幾乎不花甚麼時間

(4) 我程式最好就是 0.1，因為我程式的流程是先根據 hardblock 的形狀去做 initial floorplan，當我沒辦法找到一個可以 initial 的 floorplan 那也就沒辦法繼續往下了。

(5)



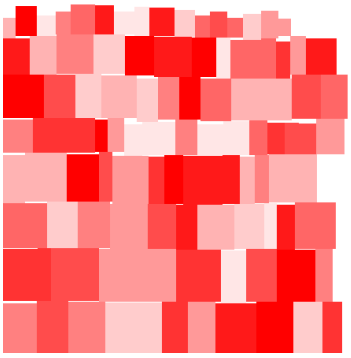
(6) 我沒有做甚麼特別的去改善時間，因為我不是使用完整的 SA，我在找 initial 的 floorplan 其實時間就花的不多，而且已經是一個可以接受的答案了，之後為了維持 outline 和 non overlap，能做的也不多，主要都在研究怎樣才能在維持結構的前提下改善 wirelength。

(7) 我的程式時間花的並不多，因為我為了有把握的維持 outline 和 non overlap，在 initial floorplan 後，我並沒有做太大的變動，所以其實能調的範圍有限，一次調整不是橫向的 hardblock 交換，就是縱向的 hardblock 交換，雖然整體看起來和自由調整沒差，但是很容易進到進步幅度慢的地方，那程式就會認為已經夠好了，那所得到的 wirelength 其實就沒這麼好，我覺得應該可以再把調整的順序做一些其他的組合，或是記錄其他資訊，讓程式可以隨時維持還不錯的改進量，那也許有助於改善 wirelength。

(8) 這次作業做下來，我發現要驗證演算法的效率其實並沒麼容易，因為像這次作業，測資內的 hardblock 其實沒有太多奇怪的形狀，也就是說其實按照一定規律去擺，通常都可以得到還不錯的答案，如果 hardblock 的大小差距很大，這會使得光是找 initial 的 floorplan 就很難了，接著用 SA 去 perturb 的話，所得出的結果也未必和 initial 有多大差異。

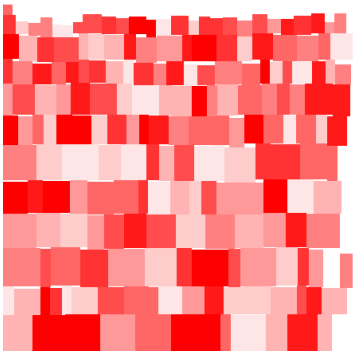
而這次作業我覺得比較難的是找 initial 的 floorplan 和在調整 floorplan 時，

要維護紀錄 floorplan 的資料結構，找到可以 swap 的兩個 hardblock，畢竟在做的時候，perturb 的時候並不會知道這個 perturb 是否會被接受，也不知道 T 是否降到趨近零，也就是要維持不超出 outline，而且在 swap 之後還要能夠調整 floorplan 避免 overlap，理由跟維持 outline 一樣，這使得 perturb 要做很多事，也就會形成瓶頸。

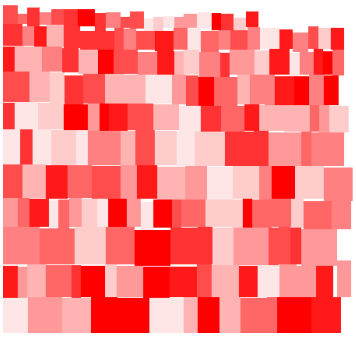


1. n100 0.1

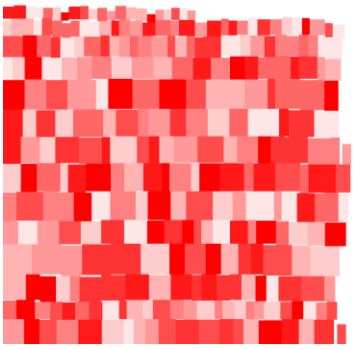
2. n100 0.15



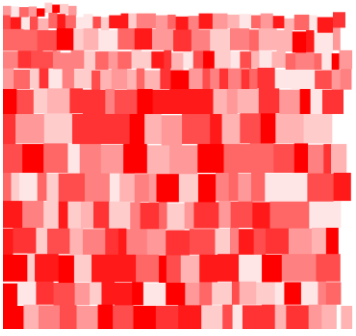
3. n200 0.1



4. n200 0.15



5. n300 0.1



6. n300 0.15