

Homework 4: Global Routing

314553017 蕭希敏

Result

grading on 314553017:

checking item	status
correct tar.gz	yes
correct file structure	yes
have README	yes
have Makefile	yes
correct make clean	yes
correct make	yes

testcase	overflow	wirelength	runtime	status
public1	0	7093	0.17	success
public2	0	306092	1.95	success
public3	0	1036142	44.15	success
public4	0	2293876	204.61	success

Successfully write grades to HW4_grade.csv

Method

主要流程為：Parsing -> Pattern Routing(L-shape) -> 檢查Overflow -> 若有Rip-up -> 更新cost -> Re-route(By A*) -> 重複到 Overflow=0 或超時

Pseudocode:

```
Parse input -> build grid size, capacities, and nets
Initialize edge usage and history
for each net:
    L-shape route (x-first and y-first), choose lower cost
Compute total overflow
while overflow > 0 and not timeout:
    mark overflow edges
    for each net:
        if net uses overflow edge:
            rip up its path
    increase history penalty on overflow edges
    reroute ripped nets with A* using updated edge costs
```

```
recompute overflow  
Output total wirelength and segments
```

1. L-shape routing 的實作是計算先走 x 再走 y，以及先走 y 再走 x 的Manhattan Distance，然後計算兩條路徑的cost（基於邊的congestion與歷史權重，但目前歷史權重皆為0），選擇cost較低者作為初始路徑，並更新 edge usage。cost公式:

$$\text{cost} = 1 + \text{congestionWeight} * (\text{usage} / \text{capacity}) + \text{historyWeight} * \text{history}$$

congestionWeight預設為4

2. A* Search 以 Manhattan distance 作為 heuristic，並每一步用usage、capacity更新cost，cost的計算公式如上。

usage會把本次要走的 net 加 1，history會在 overflow 邊上逐步累加，讓之後routing時更容易避開持續擁塞的邊。

Rip-up and Re-route

我有實作 rip-up and re-route。首先先找出所有 overflow 的邊，然後掃過所有 nets：只要該 net 的路徑經過任何一 菇overflow 邊，就把該 net 的路徑rip-up，並把它加入re-route清單。re-route的順序採用原本的 net 順序依序跑 A*。

What have you learned from this homework? What problem(s) have you encountered in this homework?

這次作業有讓我更理解global routing的實作，因為這次的目標是將overflow降為0，感覺rip-up 和 re-route的設計也特別重要，後面也花比較多時間在那，實作時也遇資料結構設計不佳的問題，像一開始我用每條 net 存所有邊的list，每次 rip-up 或re-route都要線性掃描整條路徑並更新 edge usage，這導致很費時又很容易出錯，後來靠AI給的建議，在 grid 上直接用 2D array 紀錄 hUsage/vUsage，更新只需沿路徑走一次，runtime也下降很多。