Problem B: Routing with Cell Movement

Kai-Shun Hu and Ming-Jen Yang

Synopsys, Inc.

Q&A

Q1. According to p.4, the rule of extra demand for specific cell adjacencies would be provided in the input files. However, the rule only includes cases of the adjacencies of two master cells. How should we calculate the extra demand if there are more than two master cells in the same gGrid?

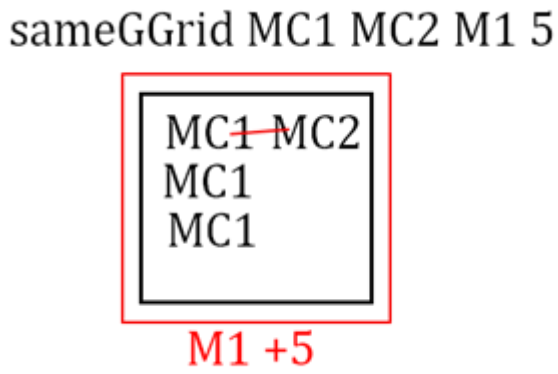
For example, if the input includes same GGrid MC1 MC2 M1 3

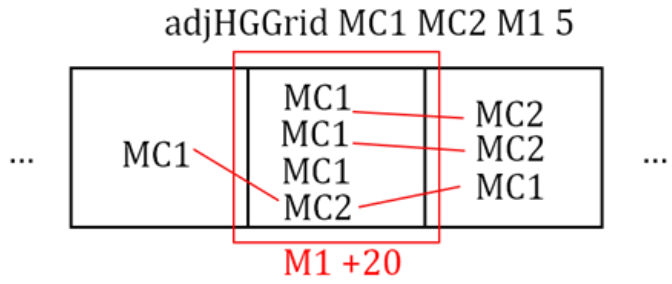
How to obtain the extra demand of a gGrid with two MC1 and two MC2?

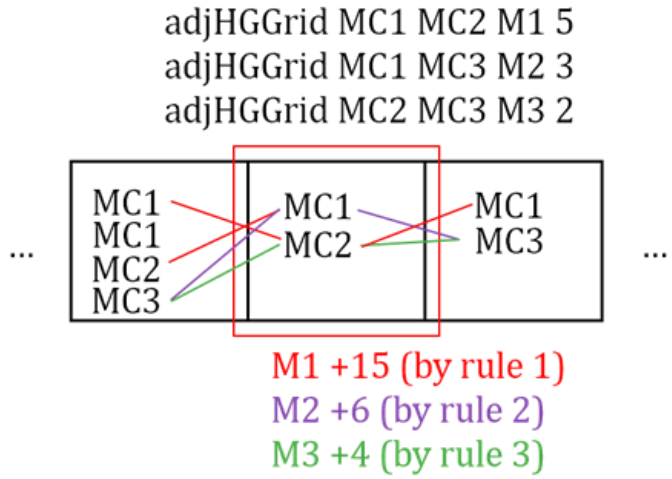
A1. Here is the explanation with examples and pseudo codes.

We calculate the extra demand by maximum master cell pairs that can be formed.

For example, for the center gGrid:







We also give pseudo code to describe how evaluator calculate extra demand.

Algorithm 1 Pseudo code describe how evaluator compute extra demands

```
for each row ∈ [1, maxRow] do
    for each col ∈ [1, maxCol] do
        CICur ← Cell Insts List placed in (row,col)
        CIPre ← Cell Insts List placed in (row,col-1)
        CINxt ← Cell Insts List placed in (row,col+1)
        MCCntCur ← Count Number of Master Cells appeared in (CICur)
        MCCntPre ← Count Number of Master Cells appeared in (CIPre)
        MCCntNxt ← Count Number of Master Cells appeared in (CINxt)
        for each rule ∈ sameGridExtraDmdRule do
            (MC1, MC2, layer, dmd) ← definition of rule
            PairCnt ← min(MCCntCur[MC1], MCCntCur[MC2])
            Add PairCnt × dmd demands to GGrid (row,col,layer) for rule
        for each rule ∈ adjHGridExtraDmdRule do
            (MC1, MC2, layer, dmd) ← definition of rule
            if MC1 = MC2 then
                PairCntPre ← min(MCCntCur[MC1], MCCntPre[MC1])
                PairCntNxt ← min(MCCntCur[MC1], MCCntNxt[MC1])
            else
                PairCntPre ← min(MCCntCur[MC1], MCCntPre[MC2])+min(MCCntCur[MC2], MCCntPre[MC1])
                PairCntNxt ← min(MCCntCur[MC1], MCCntNxt[MC2])+min(MCCntCur[MC2], MCCntNxt[MC1])
            Add (PairCntPre + PairCntNxt) × dmd demands to GGrid (row,col,layer) for rule
```

Last,

for rule sameGGrid MCX MCY MZ D

We guarantee that the x != y.

For rule same/adjHGGrid MCX MCY MZ D

We guarantee that the pair (x,y,z) will not repeat in the input.

Also, the pair (y,x,z) will not co-exist with (x,y,z) in given input.

Q2. Consider the cases that two pins that are supposed to connect to each other are in the same gGrid. Does the connecting wire also increase the demand in that gGrid? Besides, how should we write such wires in the output file in this case?

A2. Yes, it will increase the demand by 1.

You can imagine as that every pin will automatically consume 1 routing demand to the gGrid it located(no matter how much same net pin in one gGrid) since we ask that all pin must be connected.

The two or more pins in same gGrid (row,col,lay are all equivalent) are considered connected by nature but still consume 1 demand.

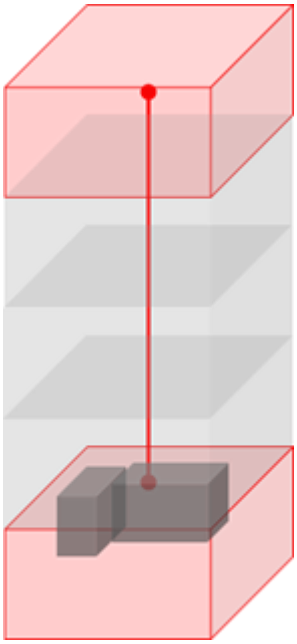
Contestants do not need output net segments(routes) with same start/end gGrid.

However, a special case that if a net has min routing layer constraint.

Even two or more pins are in the same gGrid, contestants must output a wire from pin location to the gGrid above which on min routing layer. (please see example in picture)

Otherwise it will be considered as open.

If the two or more pins are all located equal to or above min routing layer, then this wire is not needed.



For easier understanding, you can imagine that every net with min routing layer constraint has duplicated pin locate on the gGrid above on min routing layer (if the pin is under min layer).
And you must connect all pins and duplicated pins.

Q3. Would the rules of input files follow the orders in the instruction?
A3. Yes, the section order in the input file is guarantee. Please see released case1~3.

Q4. Is there a maximum of the number of gGrid?
A4. The maximum number of rows will be <= 2000.
The maximum number of columns will be <= 2000.
The maximum number of layers will be <= 32.

Q5. The question about the benchmarks of problem B (Routing with cell movement). On the problem description, it is mentioned that the routing direction is always horizontal on M1. However, on case3 benchmark, the routing direction of M1 is vertical. How should we handle the benchmark in this case? It seems the initial routing and the evaluator are still assuming M1 is horizontal.
A5. M1 would always be horizontal. The content in the released case3 were incorrect.
We have provided the new version of case3 to correct this issue.

Q6. we want to know if there will be overflows in the initial solution that you gave.
A6. No. The initial solution would meet all constraints.
i.e. would not have overflow in the initial solution.

Q7. How do you determine if we use compilers other than gcc and g++ since we're not required to provide our source code? For example, participants using Intel C compiler (ICC) may take great advantage of ICC's superior single-instruction-multiple-data (SIMD) support. Our question is, how can the rules be imposed upon all participants such that no one will take advantage of this situation?
A7. From our perspective, as long as the submitted binary can be executed on the given evaluation

environment, we should accept this binary.
(Unless this contest has the rule to forbidden the commercial tool usage. User needs to purchase license to use Intel C Compiler.)
Even in gcc, there is the optimization level. If user does not specify the optimization level when they compile their program by using gcc, it is their own choice as well.