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?

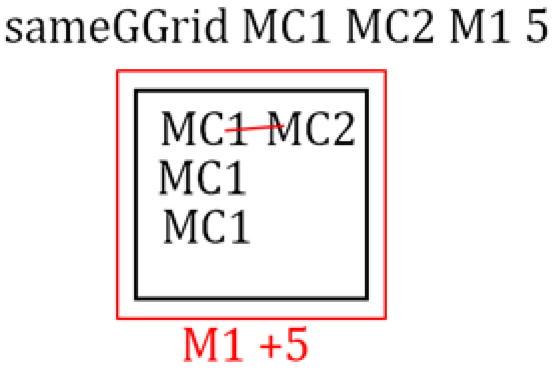
For example, if the input includes same GGrid MC1 MC2 M1 3

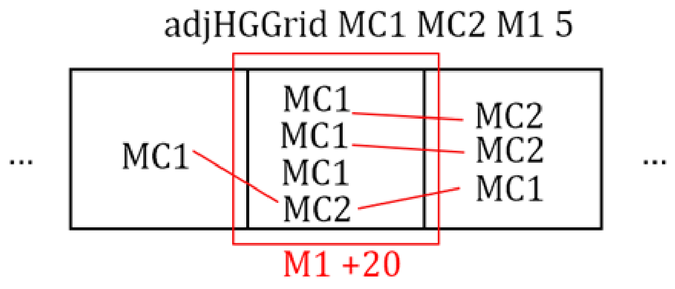
How to obtain the extra demand of a gGrid with two MC1 and two MC2?

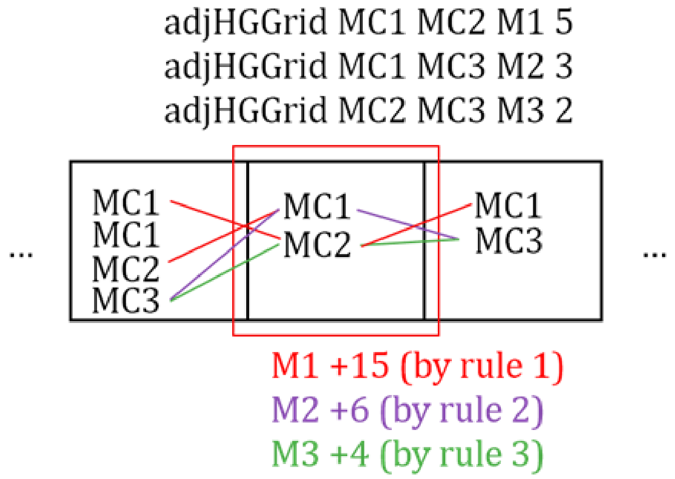
A1. Here is the explanation with examples and pseudo codes.

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

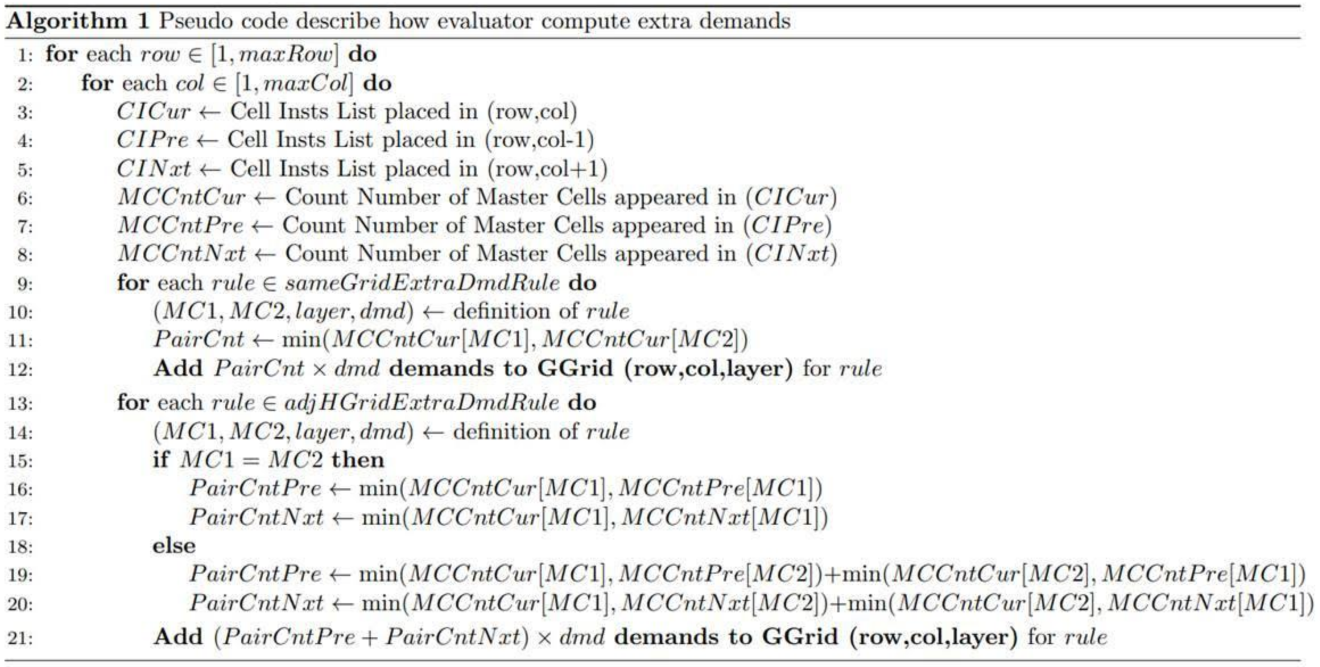
For example, for the center gGrid:







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



Algorithm 1 Pseudo code describe how evaluator compute extra demands

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

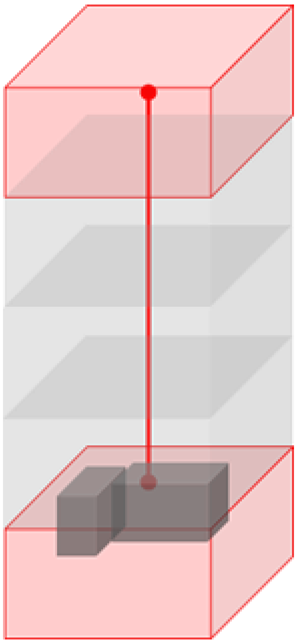
Contestants do not need output net segments(routes) with same start/end gGrid.

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.

Q8. I have some questions about the evaluator and benchmarks that are released for the contest (Problem B, case1 and case2):

(1) For case1, I have written an evaluator that calculates the wire length of the initial routing which is provided. I calculate wire length based on the number of gcells that each segment spans. If this is the right assumption, the wire length that my evaluator calculates is different from the evaluator which uploaded on the website. For example, for benchmark case1.txt, I calculate 58 units for wire length, where the contest evaluator shows 64 units for initial routing.

(2) In case2, the initial routing that is provided is not valid. When I use the evaluator shows that it is not valid routing. Is it possible that the initial routing that is provided by the contest to be not valid?

A8. (1) For Q1, we checked our evaluator result again and believe our result (i.e. 64 units for case1 initial routing) is correct.

Contestants can use "./evaluator case1.txt -verbose" to know the routing length of each net.

If contestants still think the released evaluator result is incorrect, please provide us the net name with correct length result. We will double check it again.

(2) For Q2, the released data for case2 is incorrect. Thanks for pointing this out.

We have correct it for case2. Please help to update the "cases.tgz" to the website.

We guarantee the provided initial routing must be valid.

Q9. Q&A-Q6 Says: 'would not have overflow in the initial solution'

But I found that Case3 Grid (14 4 2) is -1 after all demand checking.

Is it wrong? or I misunderstood the giving data?

Attachment is the part of the case3 which will change the supply in that grid.

A9. adjHGGrid demand is for adjacent "horizontal" gGrids.

And, the data format is with the "row" first then followed by the "column". In your example, it is row=13 col=4 with row=14 col=4. Which is vertical. So, there is no adjHGGrid demand in this case.

```
adjHGGrid MC222 MC129 M2 1
    CellInst C2296 MC129 13 4 Movable
    CellInst C1157 MC64 14 4 Movable
```

//9

By the way, you can use -verbose with the released evaluator. (i.e. "./evaluator case.txt -verbose")

With -verbose option, the evaluator would print out all the gGrids which need adjHGrid and sameGGrid demands. That can help you debug/validate your result.

Q9. How we calculate the segment demands of one GCell? Are we considering the via segments as an extra demand?

For example, if we have the following segments passing through GCell (2,3,1):

Seg1: 1 3 1 2 3 1

Seg2: 2 3 1 2 3 2

According to the above sample the demand of GCell (2,3,1) would be one (just counting the vertical routed) or would be two (counting routed wire and via segment)?

If we consider the via demands of the cells, many of GCells according to my evaluation are overflowed.

A9. For a specific net, no matter how many segments of that net passing through a certain gGrid, we will only count one demand.

So in your case, as long as the seg1 and seg2 are segments of the same net, the gGrid (2,3,1) will only count one demand.

Take another example:

Seg1: 1 3 1 2 3 1 N1

Seg2: 2 3 1 2 3 2 N1

Seg3: 2 3 1 3 3 1 N1

In this case the demand of gGrid (2,3,1) is still counted one for net N1.

We will update the released evaluator soon to provide more detail information of how the demand of specific gGrid is calculated.

Q10. I am wondering if you can guide me how the demand of each GCell is measured. According to my understanding, we have three types of demands:

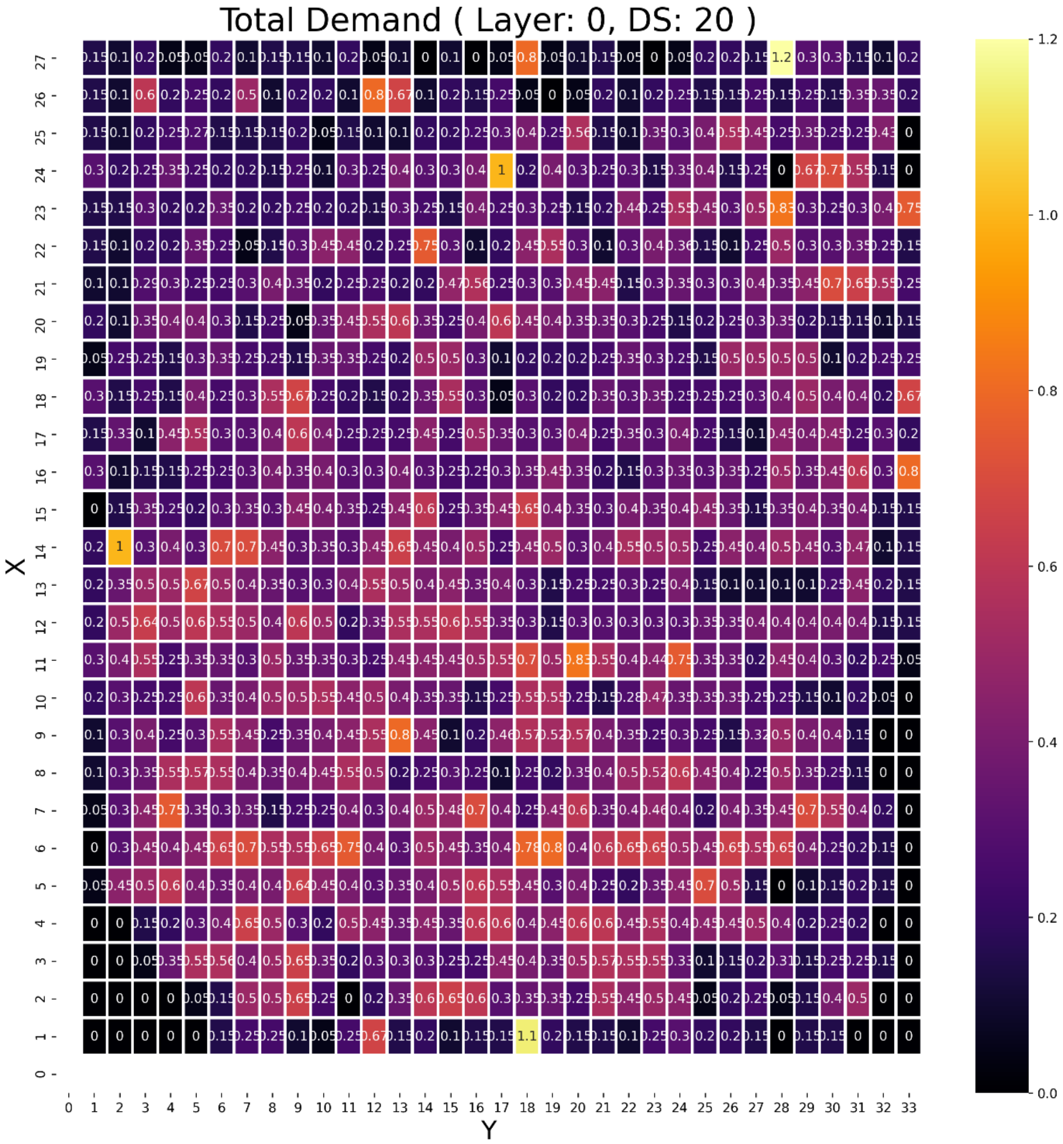
1. Blkg Demand
2. Wire Demand
3. Adjacency Demands

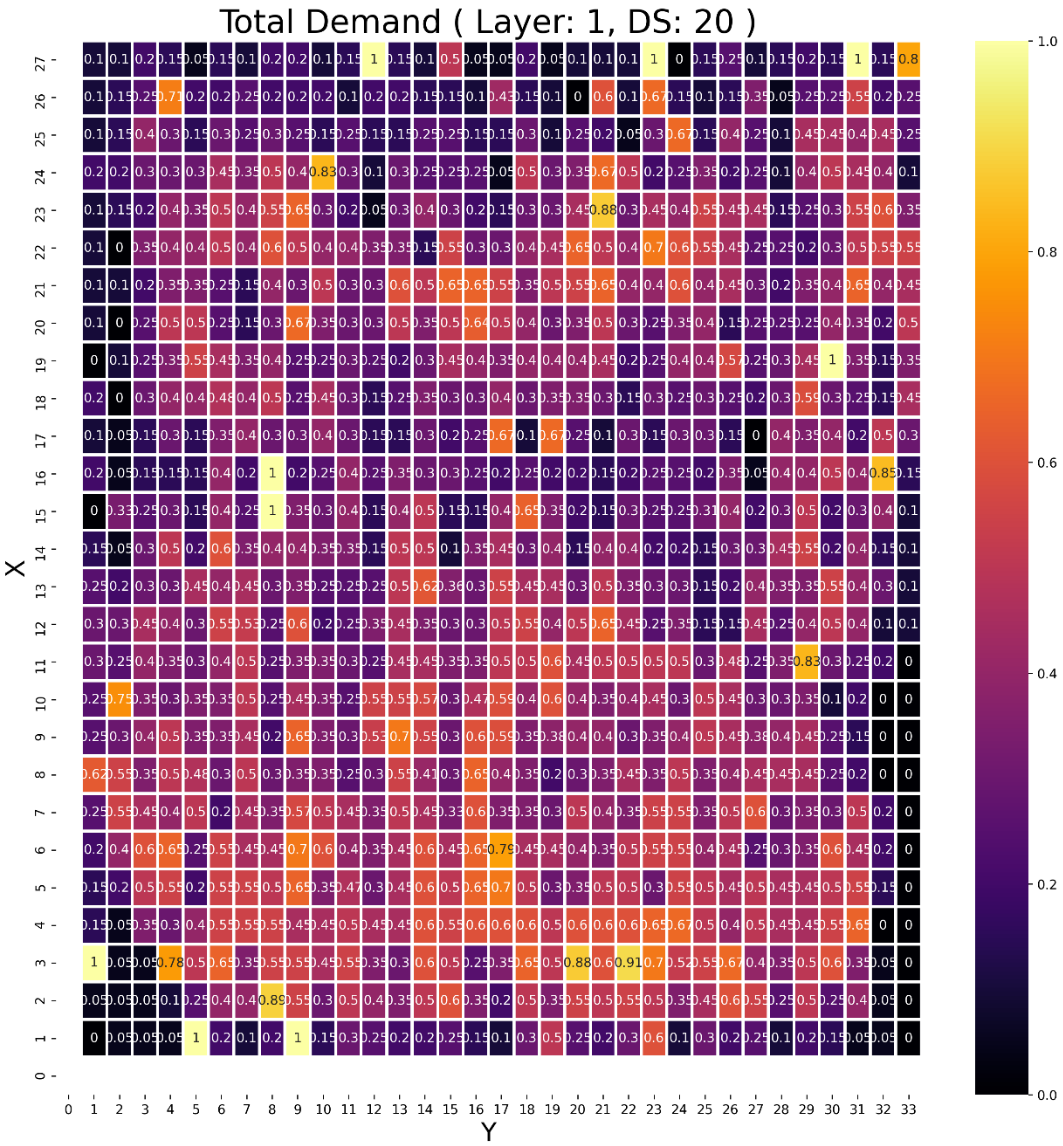
After implementing the demands and normalise it to the supply of each gcell, I figure out there are a bunch of gcells overflowed which is opposite of the contest claim "The initial solution does not have any overflow". I attached the congestion image of each layer for case3 benchmark.

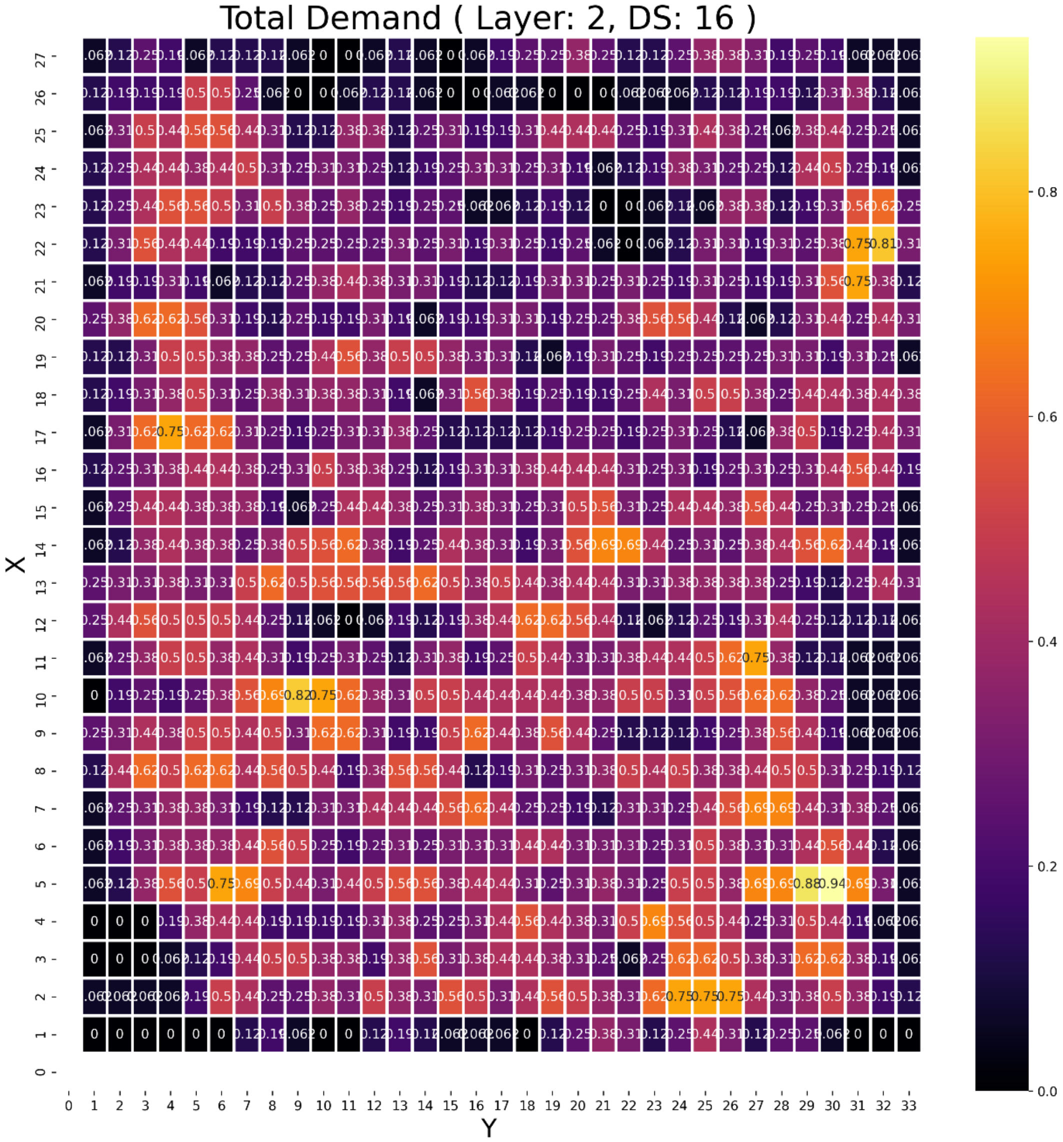
The demand of each gcell is calculated as follow:

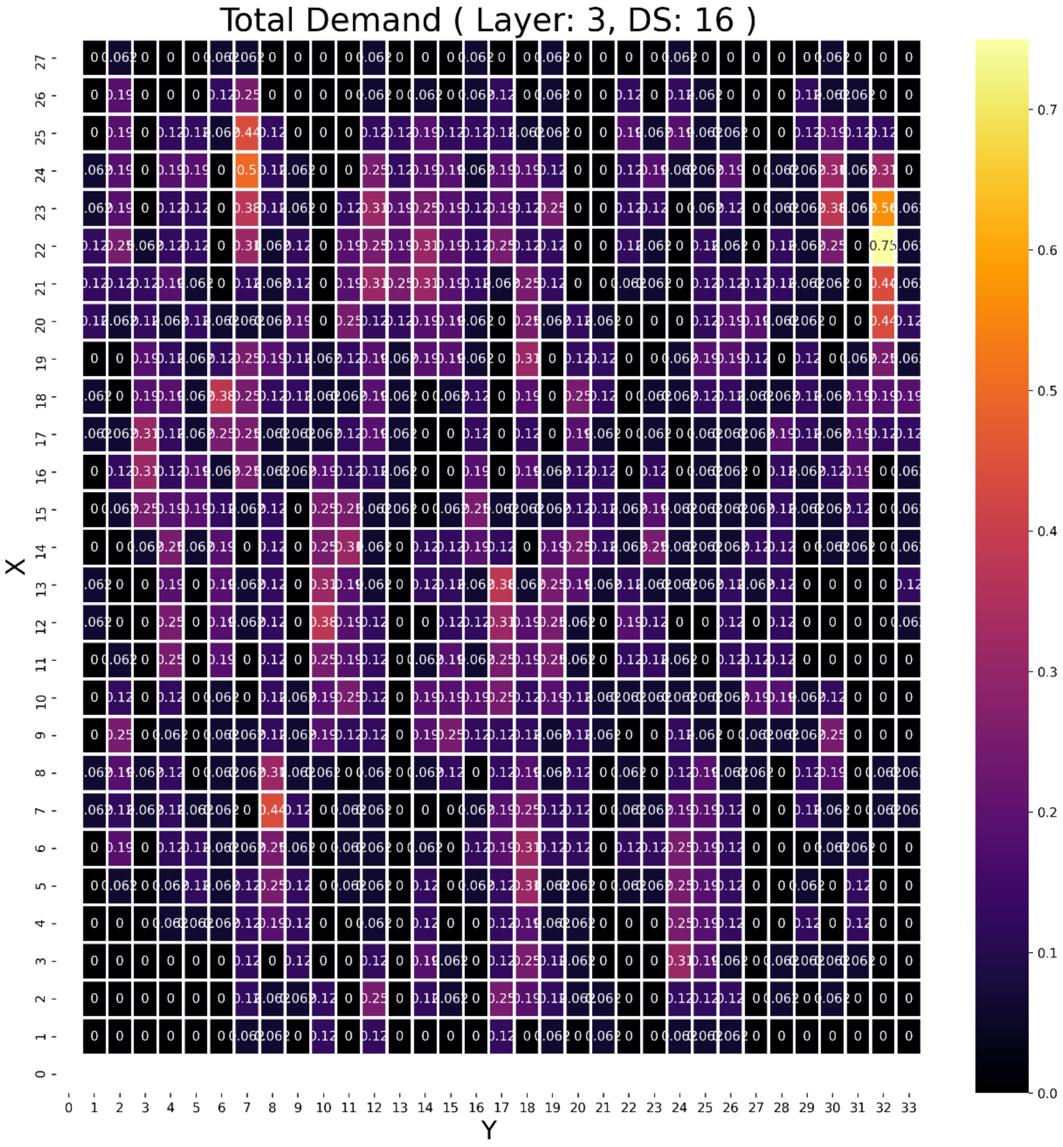
Total_demand(i,j,z) = (demand_blkg(i,j,z) + demand_wire(i,j,z) + demand_adjacency_cell(i,j,z)) / gcell_supply(i,j,z)

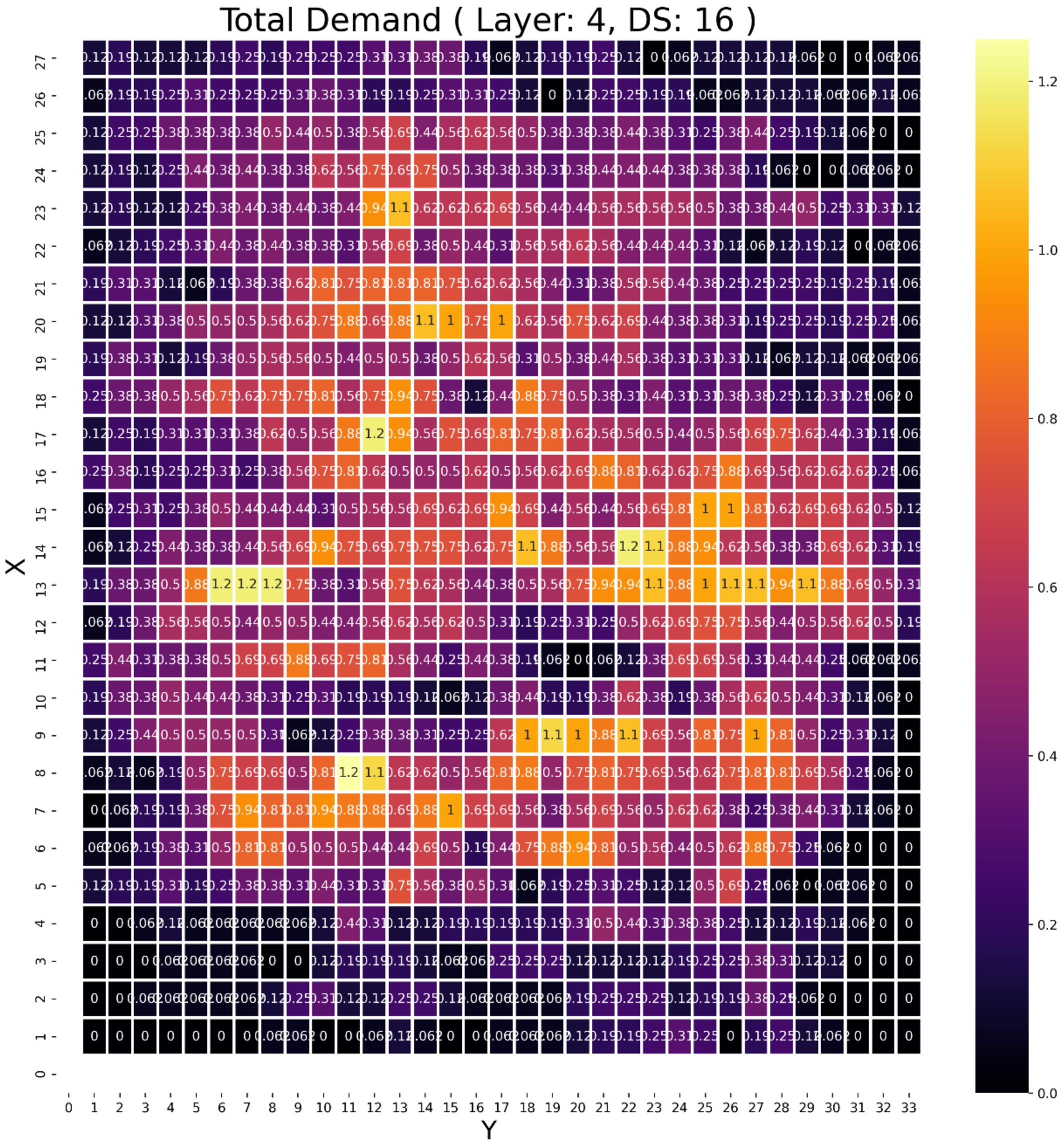
I am wondering if my assumption is completely wrong in the total_demand calculation of one gcell.

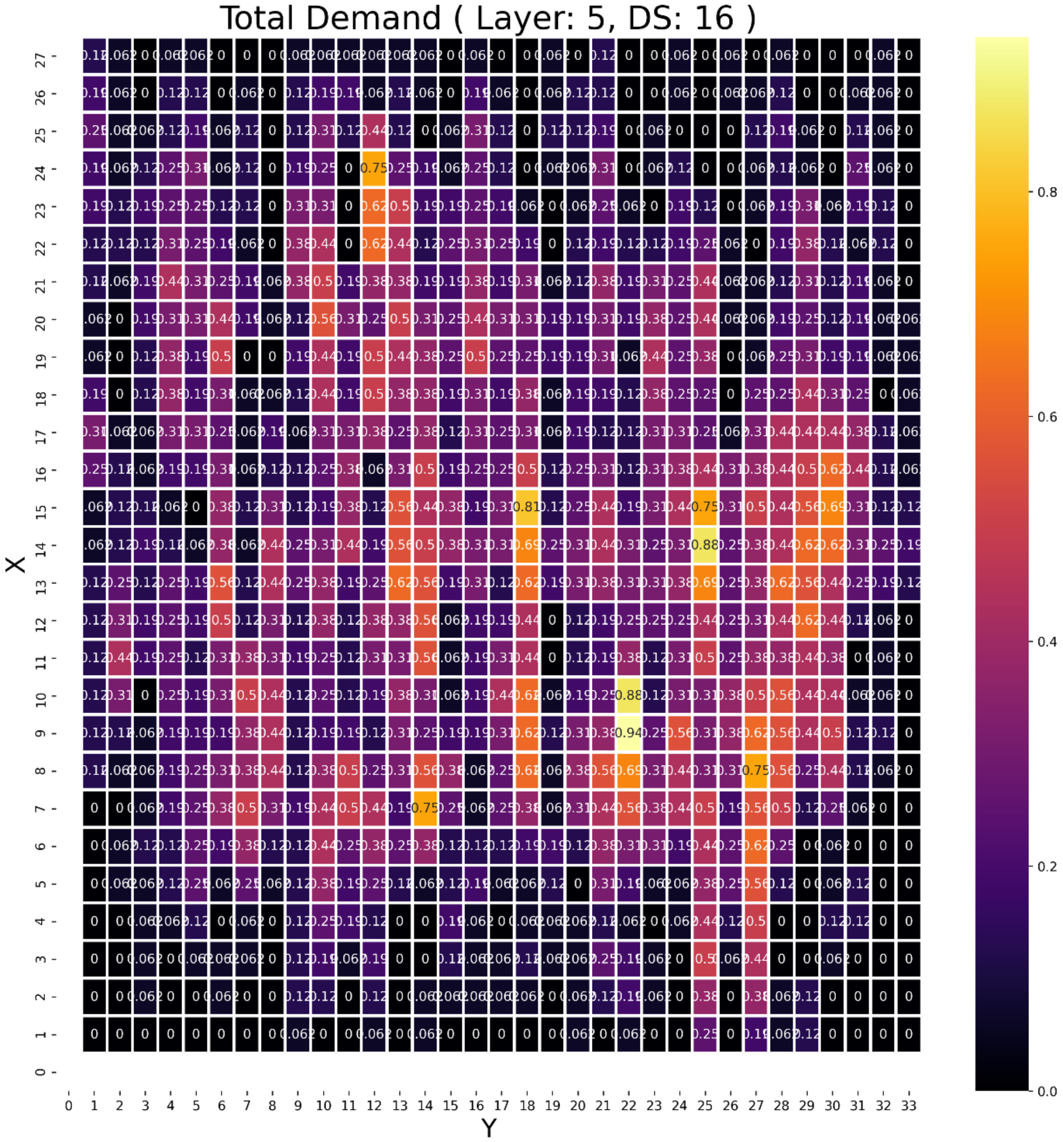

Total Demand (Layer: 0, DS: 20)
X
Y
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
0.0
0.2
0.4
0.6
0.8
1.0
1.2

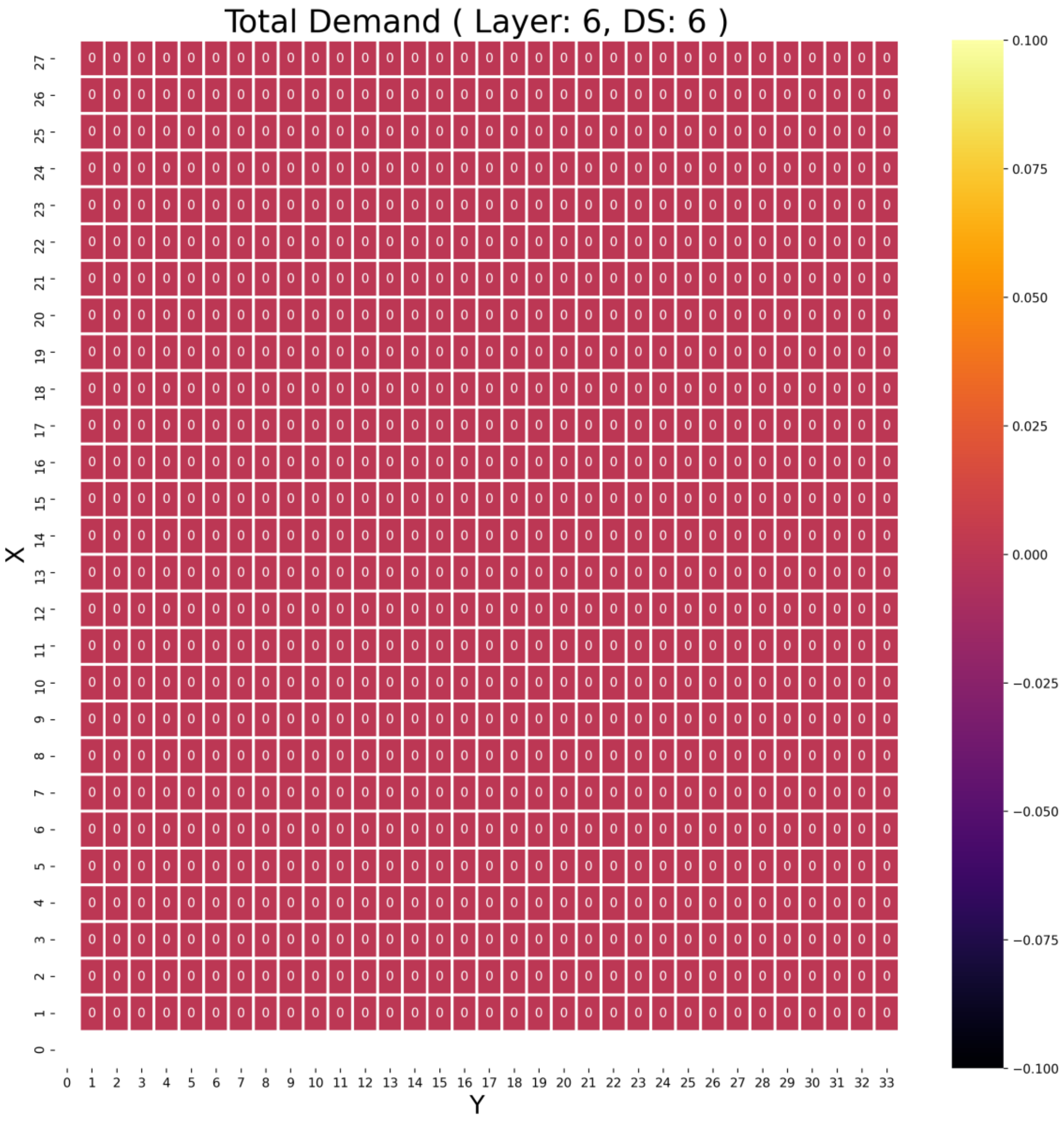

Total Demand (Layer: 1, DS: 20)
X
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
Y
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
1.0
0.8
0.6
0.4
0.2
0.0


Total Demand (Layer: 2, DS: 16)
X
Y
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
0.0
0.2
0.4
0.6
0.8


Total Demand (Layer: 3, DS: 16)
X
Y
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7


Total Demand (Layer: 4, DS: 16)
X
Y
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
0.0
0.2
0.4
0.6
0.8
1.0
1.2


Total Demand (Layer: 5, DS: 16)
X
Y
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
0.0
0.2
0.4
0.6
0.8



A10. We don’t see the overflow by our evaluator in the gGrid in your provided attachment. We also updated the evaluator so now you can indicate the debug gGrid and the evaluator will show you all the demand it calculated.

For example in (13,6,5)

```
./evaluator case3.txt --dbgrow 13 --dbgcol 6 --dbglay 5
dbgGGrid total supply:16
dbgGGrid found route dmd 1 by net N146. current total dmd:1
dbgGGrid found route dmd 1 by net N239. current total dmd:2
dbgGGrid found route dmd 1 by net N241. current total dmd:3
dbgGGrid found route dmd 1 by net N296. current total dmd:4
dbgGGrid found route dmd 1 by net N299. current total dmd:5
dbgGGrid found route dmd 1 by net N326. current total dmd:6
dbgGGrid found route dmd 1 by net N370. current total dmd:7
dbgGGrid found route dmd 1 by net N371. current total dmd:8
dbgGGrid found route dmd 1 by net N698. current total dmd:9
dbgGGrid found route dmd 1 by net N762. current total dmd:10
dbgGGrid found route dmd 1 by net N850. current total dmd:11
dbgGGrid found route dmd 1 by net N870. current total dmd:12
dbgGGrid found route dmd 1 by net N969. current total dmd:13
```

```
./evaluator case3.txt --dbgrow 1 --dbgcol 18 --dbglay 1
dbgGGrid total supply:7
dbgGGrid found route dmd 1 by net N1805. current total dmd:1
dbgGGrid found route dmd 1 by net N1923. current total dmd:2
dbgGGrid found route dmd 1 by net N1974. current total dmd:3
dbgGGrid found route dmd 1 by net N1981. current total dmd:4
dbgGGrid found route dmd 1 by net N1991. current total dmd:5
dbgGGrid found pin blkg dmd 1 by CellInst775. current total dmd:6
dbgGGrid found pin blkg dmd 1 by CellInst820. current total dmd:7
dbgGGrid found pin blkg dmd 0 by CellInst2666. current total dmd:7
```

Please let us know the detail of each demand you calculate for a specific grid to verify if the provided case is overflown.

Q11. I have one more question, there are some nets in case3.txt for example N1946 or N1708 that there is no initial routing for them. Is this a valid initial solution provided by contest or it is a bug in the case3.txt benchmark?

A11. These nets are the net that all pins in the same gGrid (row=2 col=6 layer=1 in this case).
We doesn't needs to report segment starts and ends in the same gGrid.
These nets are considered naturally connected, no routing needed. But, it still consumes 1 demand for this gGrid for this net.

Q12. Could you please answer these trivia questions about the Problem B input format for us?
Can we assume all layers are named by M*i*, Master cells are named by MC*j*, cells are named by C*k*, and nets are named by N*x*, where *i* = [1, #layer], j = [1, #Master cell], *k* = [1, #cell], and *x* = [1, #net]?
And in each Master cell, can we assume pins are named by P*y* and blockages are named by B*z*, where *y* = [1, #pin in this MC], and *z* = [1, #blockage in the same MC]?
Since if so, we can save a little string storage and table look-up.

A12. All these assumptions in your email are valid. You can have all these assumptions in your program.