Problem B: Routing with Cell Movement Advanced

Kai-Shun Hu, Ming-Jen Yang, and Tao-Chun Yu

Synopsys, Inc.

0. Revise History

Feb/19 – initial version

1. Introduction

Splitting the physical implementation problem to 2 sub-problems (placement and Routing) is a common approach to solve the modern complex physical implementation problem. Inside these 2 major steps, the common approach is to further divide the placement/routing problem into more sub-problems. The purpose of this divide-and-conquer approach is to make sure this complex problem would be manageable and be converged in reasonable runtime.

With more and more sub-problems be defined, the correlation/objective of each sub-problem might not be aligned with each other. Also, in certain sub-problem, it may preserve a margin on purpose for helping the convergence of upcoming steps. This kind of margins and/or mis-alignments would become a weakness point in this divide-and-conquer approach.

For example, in the placement stage, a placer commonly constrains the placement result by cell density or pin density. It is a very conservative way in order to preserve more margins for helping the convergence of routing. But, cell density or pin density may not be well correlated to the real routing problem.

In this contest, we are trying to explore the approach of having a routing engine which can also do cell movement. So, the weakness of the preserved margins and mis-alignments issues can be eliminated.

2. Contest Objective

The objective of this contest is to develop a global route routing engine to honor all the given routing constraints. And, this global route routing engine can move the cells from one gGrid to another if all the given routing constraints can still be satisfied while the wire length can be further reduced through the cell movements.

To simplify the problem, an initial routing result would be given in the input file. Contestants can start from the given initial routing data and then do the incremental routing for re-connecting the broken parts which are caused by cell movements.

Contestants can also discard all the given initial routing data and generate all the new routings while considering cell movements by their own.

In this contest, the total number of moved cells among all cells are constrained. Having this cell movement constraint is because that we do not want the contestants to perform a whole new placement first to generate a totally different placement result. It is not the purpose of this contest.

3. Problem formulation, Input/Output Format

The input of this problem is a placed and routed design. The goal is to further minimize the routing length of the design with given constraints.

Multi-threading is allowed and encouraged. The max number of cores that can be used in provided machine is to be announced.

3.1 Input data

- In this contest, we limit the number of cell movements. So, the maximum number of moved cells will be given. However, the move distance of each moved cell is not limited.

Syntax

```
MaxCellMove <maxMoveCount>
```

Example

```
MaxCellMove 2
```

In the above example, the design allows to have two cell movements at most.

- The number of rows & columns of the gGrids will be given. gGrid is the minimum resolution of this problem. Cell instances and routes are all with the resolution of gGrid. All the routing layers have the same number of rows and columns.

Syntax

```
GGridBoundaryIdx <rowBeginIdx> <colBeginIdx> <rowEndIdx> <colEndIdx>
```

Example

```
GGridBoundaryIdx 1 1 5 5
```

In the above example, it means the design has 5 gGrid rows (from 1 to 5) and 5 gGrid columns (from 1 to 5).

- The total number of routing layers, the routing direction of each layer, and the default supply value of the gGrid on each layer will be given.

  The routing direction will always be horizontal on M1 and routing direction of two adjacent layers will always be different.
  In this contest, all H/V-direction routings must satisfy layer routing direction. And for the simplification, VIA(vertical interconnect access) is simply modeled as z-direction routing. Its gGrid length and demand consumption is same as routings in other direction.

  Also, the R/C characteristic is different on each layer. So, the routings on different layer would result to different power consumption. We would like to model a layer-based factor `<powerFactor>` to represent the power consumption of the routings on a layer.

Syntax

```
NumLayer <LayerCount>
Lay <layerName> <Idx> <RoutingDir> <defaultSupplyOfOneGGrid> <powerFactor>
```

Example

```
NumLayer 3
Lay M1 1 H 10 1.2
Lay M2 2 V 8 1.0
Lay M3 3 H 8 0.8
```

  In the above example, there are 3 routing layers in the design. The routing direction of M1 and M3 are horizontal and the routing direction of M2 is vertical. The default supply of one gGrid on M1 is 10. The default supply of one gGrid on M2 and M3 is 8. The power factor on M1, M2, and M3 are 1.2, 1.0, and 0.8 respectively.

- There will be gGrids with non-default supply. The increment or decrement value of the supply in certain gGrids will also be given.

Syntax

```
NumNonDefaultSupplyGGrid <nonDefaultSupplyGGridCount>
<rowIdx> <colIdx> <LayIdx> <incrOrDecrValue>
```

Example

```
NumNonDefaultSupplyGGrid 2
2 2 1 +3
1 2 3 -2
```

In the above example, the supply of the (2,2,1) gGrid will be the default supply of layer M1 plus 3. The supply of the (1,2,3) gGrid will be the default supply of layer M3 minus 2.

- The detailed information of the master cells which are used in the design will be given. It includes the master cell name, number of pins in the master cell, the pin name & pin layer of each pin, number of blockages in the master cell, and the blockage name, blockage layer, & blockage demand of each blockage.

Syntax

```
NumMasterCell <masterCellCount>
MasterCell <masterCellName> <pinCount> <blockageCount>
Pin <pinName> <pinLayer>
Blkg <blockageName> <blockageLayer> <demand>
```

Example

```
NumMasterCell 2
MasterCell MC1 1 1
Pin P1 M1
Blkg B1 M1 2
MasterCell MC2 2 0
Pin P1 M2
Pin P2 M1
```

In the above example, there are 2 master cells MC1 and MC2. MC1 has 1 pin P1 on M1 and 1 blockage B1 on M1. Blockage B1 would need 2 demands. While, MC2 has 2 pins, P1 is on M2 and P2 is on M1. There is no blockage in MC2.

- The placed cell instances and the netlist will be given. And, the given nets may

have min routing layer constraint. If a net does not have min routing layer constraint, `<minRoutingLayConstraint>` will represent with a keyword `NoCstr`. There will be cell instances which are not movable. This movable or fixed information will be described in `<movableCstr>` for every cell instance. If the cell instance is movable (i.e. can be moved to another gGrid), `<movableCstr>` will represent with a keyword `Movable`. If the cell instance is not movable (i.e. cannot be moved to another gGrid), `<movableCstr>` will represent with a keyword `Fixed`.

In the consideration of timing, we would like to model a net-based weight `<weight>` to represent the net criticality in terms of timing for the nets in critical paths. So, for the nets not on timing critical paths, the weight value of the nets will be 1.0. For the nets on timing critical paths, the weight value of these nets will be greater than 1.0. With this net-based weighted model, we would like to encourage the contestants to reduce the routing length for the nets on timing critical paths.

Syntax

```
NumCellInst <cellInstCount>
CellInst <instName> <masterCellName> <gGridRowIdx> <gGridColIdx> <movableCstr>
NumNets <netCount>
Net <netName> <numPins> <minRoutingLayConstraint> <weight>
Pin <instName>/<masterPinName>
```

Example

```
NumCellInst 2
CellInst C1 MC1 4 1 Fixed
CellInst C2 MC3 5 2 Movable
NumNets 1
Net N1 2 M2 1.5
Pin C1/P2
Pin C2/P1
```

In the above example, there are 2 cell instances, C1 and C2, which are placed in (4,1) and (5,2) respectively. And, there is 1 net, N1. Net N1 has 2 pins which are the P2 pin of instance C1 and P1 pin of instance C2. Furthermore, net N1 has min routing layer constraint that no horizontal and vertical routing is allowed below M2. And, net N1 is on timing critical paths and its weight value is 1.5.

- There will be voltage area constraints. It will have voltage area label for certain gGrid row & column pairs. Also, it will have corresponding voltage area label for certain cell instances as well. For the cell instances having voltage area label, these

cell instances can only be placed (moved) to the gGrid row & column pairs which has the same voltage area label. For the cell instances without voltage area label, all the gGrid row & column pairs in the given design can be the candidate for cell location.

Syntax

```
NumVoltageAreas <voltageAreaCount>
Name <voltageArea Name>
GGrids <gGridCount>
<gGridRowIdx> <gGridColIdx>
Instances <instanceCount>
<InstanceName>
```

Example

```
NumVoltageAreas 2
Name V1
GGrids 3
1 1
2 1
1 2
Instances 2
C1
C2
Name V2
GGrids 2
5 5
4 5
Instances 2
C5
C7
```

In the above example, there are 2 voltage areas, V1 and V2. gGrid (row=1, col=1), (row=2, col=1), and (row=1, col=2) are in the voltage area V1. Cell instance C1 and C2 are associated to voltage area V1 as well. So, cell instance C1 and C2 can only be placed (moved) in these 3 gGrids. gGrid (row=5, col=5) and (row=4, col=5) are in voltage area V2. Cell instances C5 and C7 can only be placed inside voltage area V2.

- Finally, an initial routing data is also given.

Syntax

```
NumRoutes <routeSegmentCount>
<sRowIdx> <sColIdx> <sLayIdx> <eRowIdx> <eColIdx> <eLayIdx> <netName>
```

Example

```
NumRoutes 2
4 1 1 4 1 3 N1
4 1 3 4 4 3 N1
```

In the above example, there are 2 routing segments for net N1. The 1st routing segment is a z-direction segment from layer 1 to layer 3. The 2nd routing segment is a horizontal segment from (row 4, col 1) to (row 4, col 4) on layer 3.

3.2 Supply and Demand

As described in section 3.1, the supply of each gGrid will be given. The supply of a gGrid would be the default supply of its corresponding layer with an increment or a decrement of its non-default supply.

The demand of a gGrid would be the summation of { the number of nets which has routing segment in this gGrid, all blockage demands for the cell instances in this gGrid}.

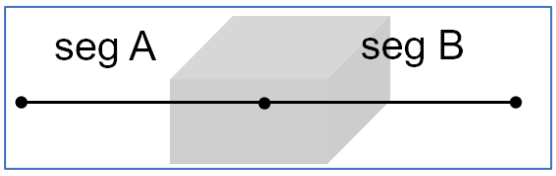
For one net, no matter how much routing segments crossing one gGrid, it will only consume one demand in that gGrid.

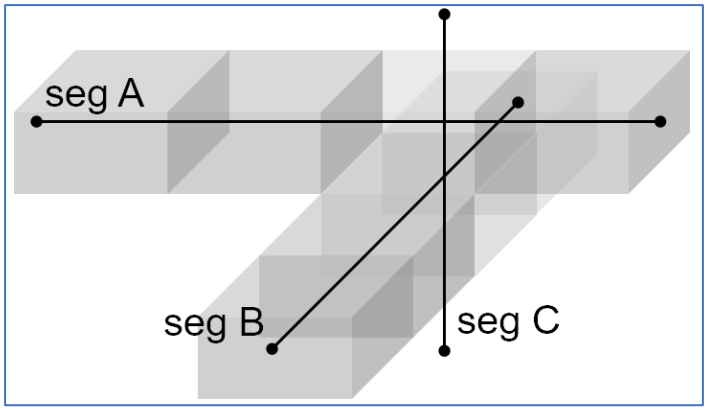
If the demand of a gGrid is larger than its supply, we call this gGrid is an overflow gGrid. In this contest, we do not allow any of the gGrid be overflow gGrid.

3.3 Connectivity model

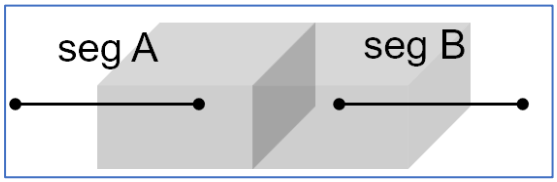
We can imagine a gGrid as a 3D block. A routing segment is considered as in the center line of the gGrid block. Two routing segments are connected if and only if they touch each other.

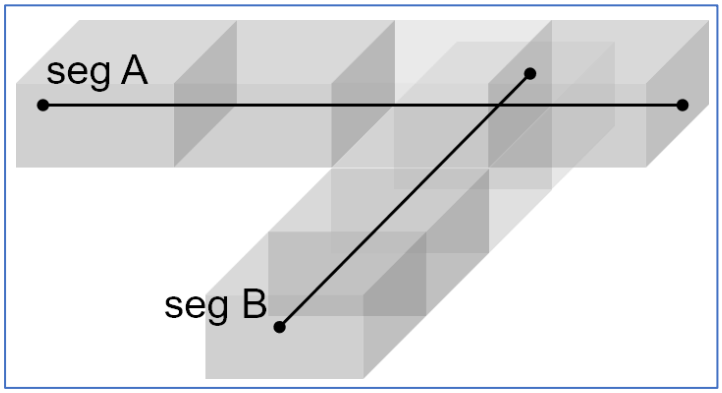
Following examples are considered as connected for same net routing segment A and segment B (and segment C).

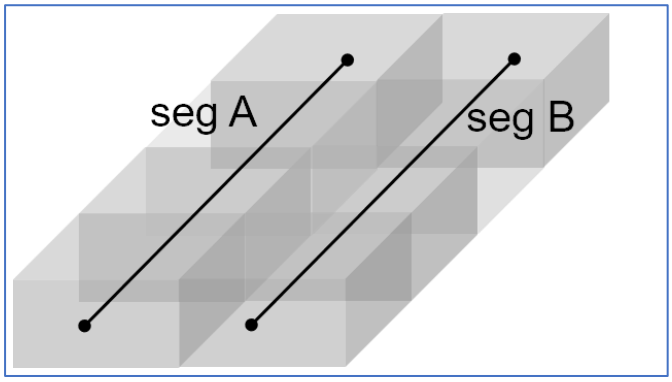




Following examples are considered as NOT connected for same net routing segment A and segment B.







For each net, all pins are required to be connected by the routing segments. Floating or dangling routing segments are allowed but they are redundant and consumes gGrid

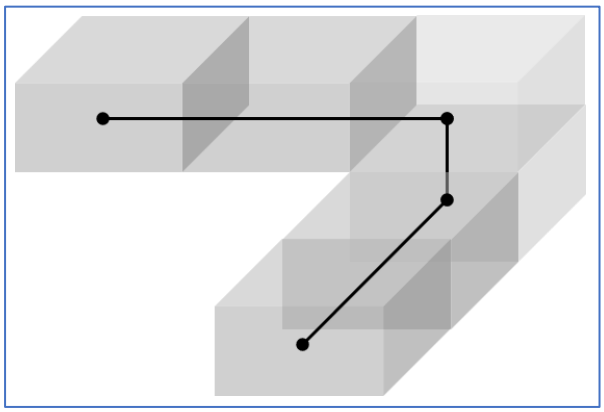
lengths.

Since this is a simplified global routing problem, contestants need not to worry about any net shorts.

3.5 Routing lengths (gGrid lengths)

The routing length of one net is defined as the number of gGrid crossed by the routing segments of that net. From other perspective, it is also equal to the demand caused by routing segments of that net.

For examples, the gGrid length of following routing is 6.



3.6 Output data

Following information is needed in the output file.

- The moved cell instances and their new locations (row and column).

Syntax

```
NumMovedCellInst <movedCellInstCount>
CellInst <instName> <newRowIdx> <newColIdx>
```

Example

```
NumMovedCellInst 2
CellInst C1 3 3
CellInst C2 4 2
```

In the above example, two cell instances are moved. Cell instance C1 is moved to (row 3, col 3) and cell instance C2 is moved to (row 4, col 2).

- No matter the given initial routing data is re-used or not, the complete routing

information needs to be written out in the output file. The syntax of output routing data is the same as the input routing data described in section 3.1.

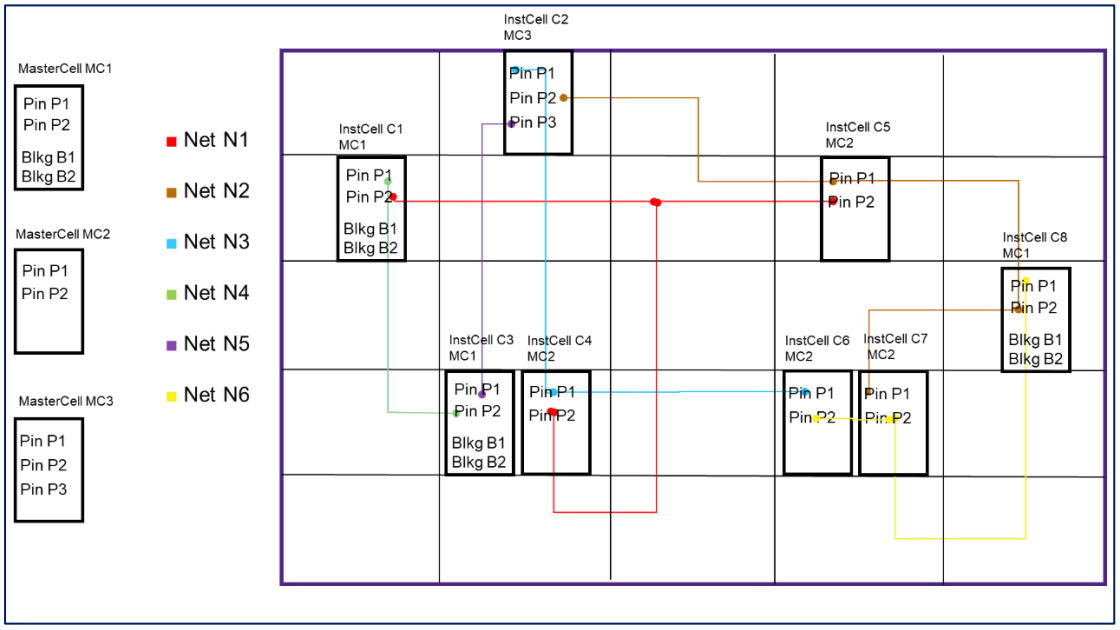
Syntax

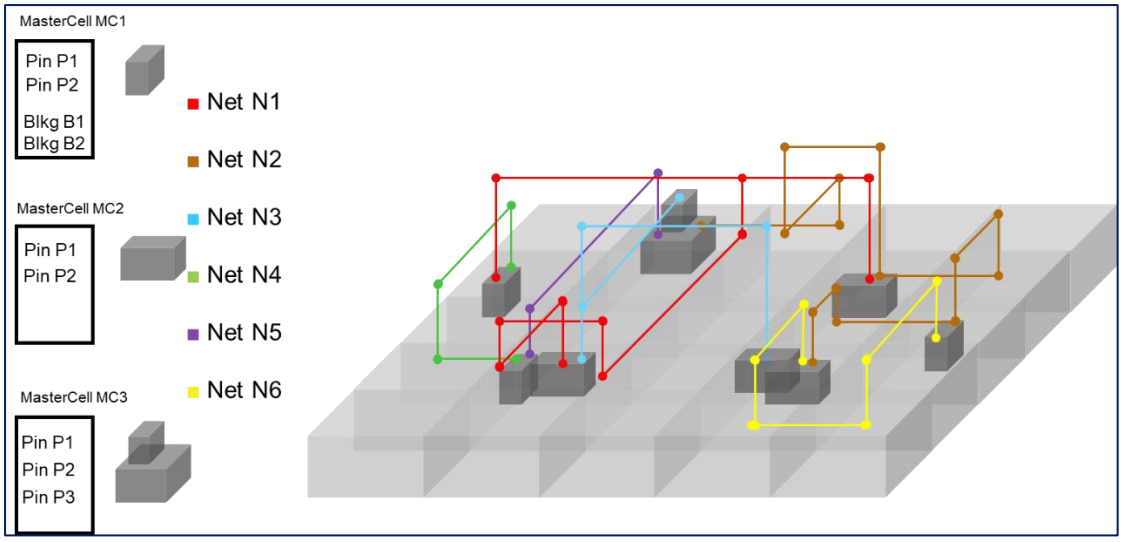
```
NumRoutes <routeSegmentCount>
<sRowIdx> <sColIdx> <sLayIdx> <eRowIdx> <eColIdx> <eLayIdx> <netName>
```

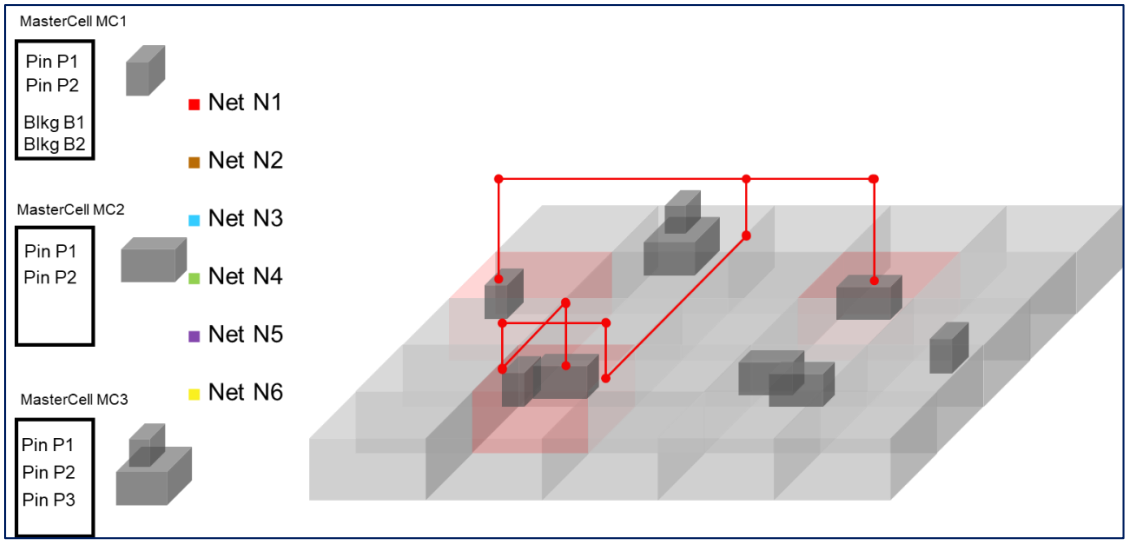
Example

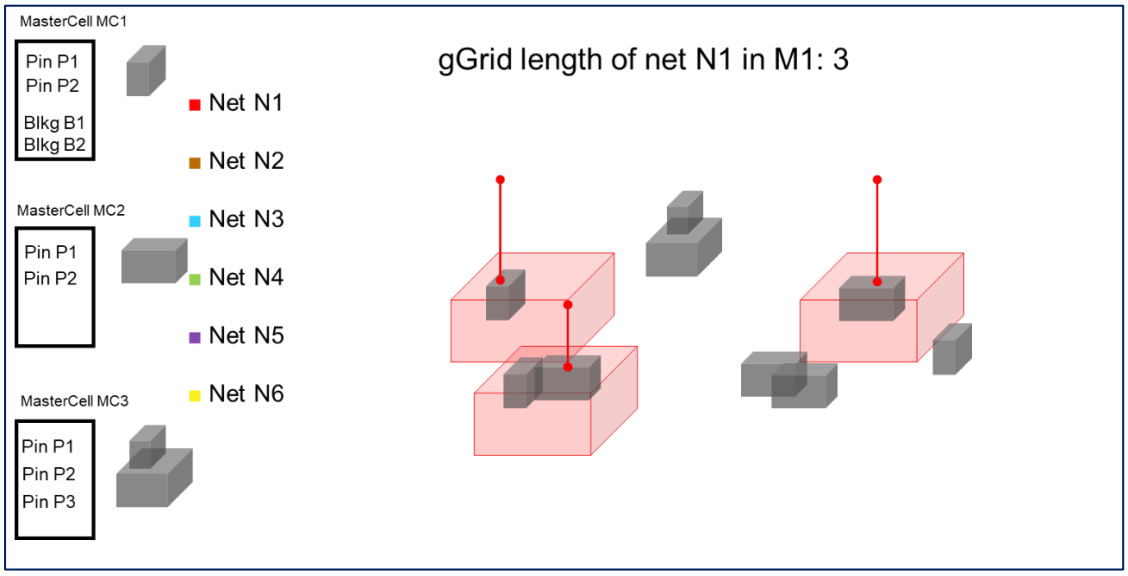
```
NumRoutes 2
4 1 1 4 1 3 N1
4 1 3 4 4 3 N1
```

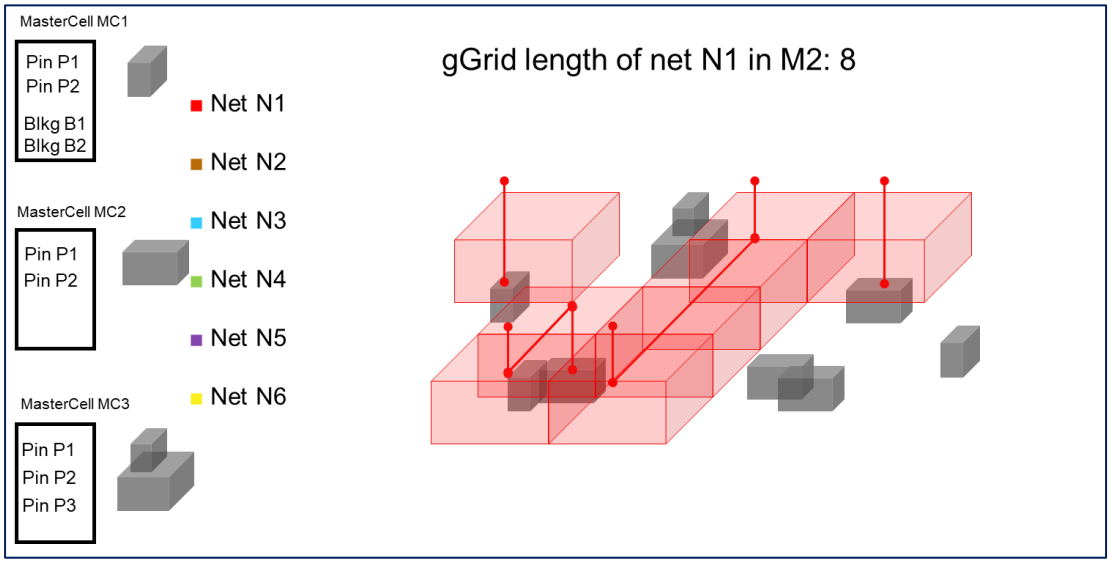
4. Example

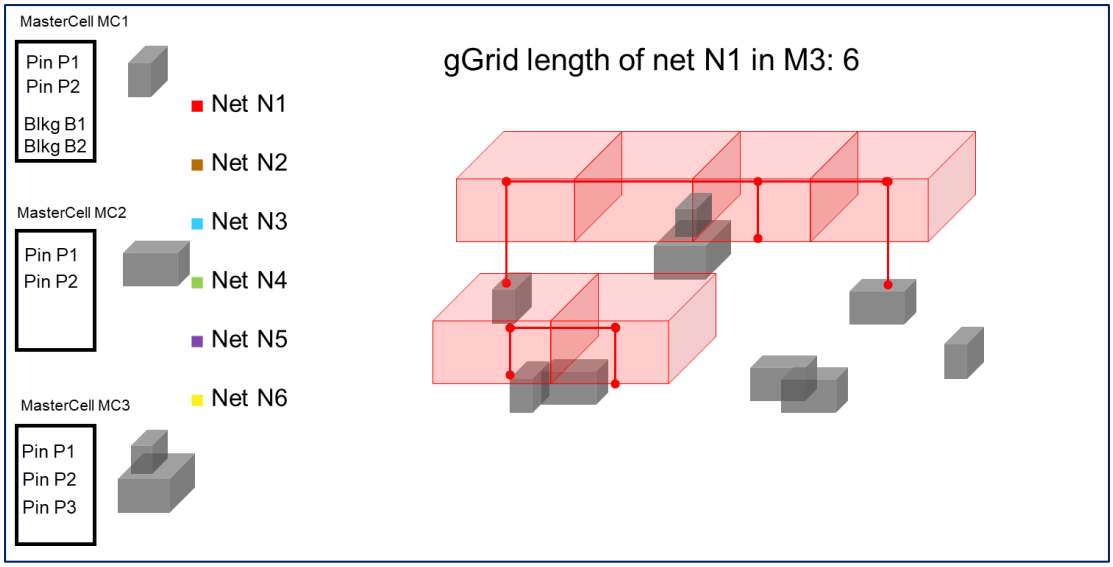


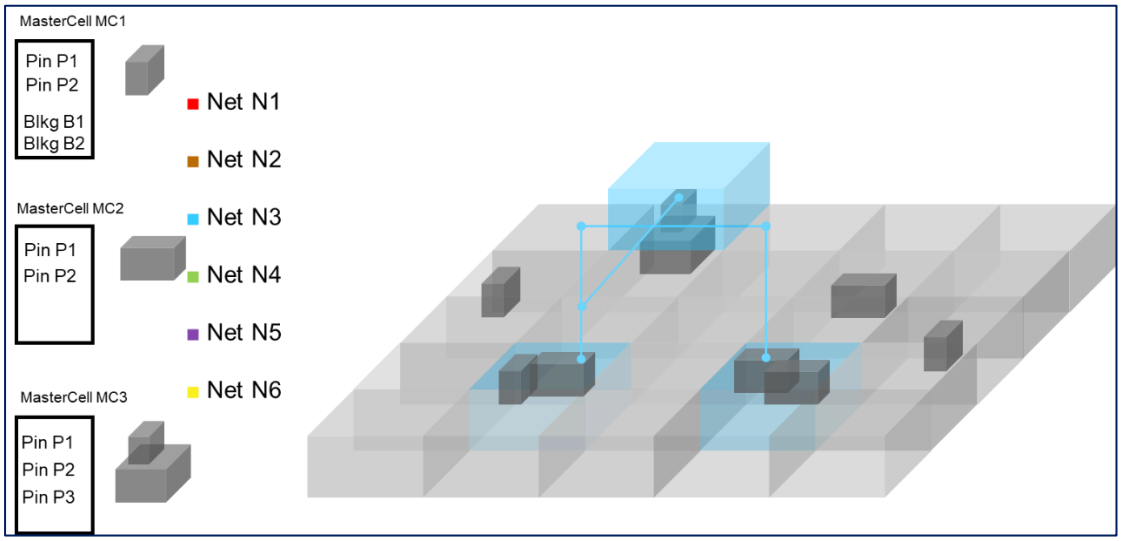


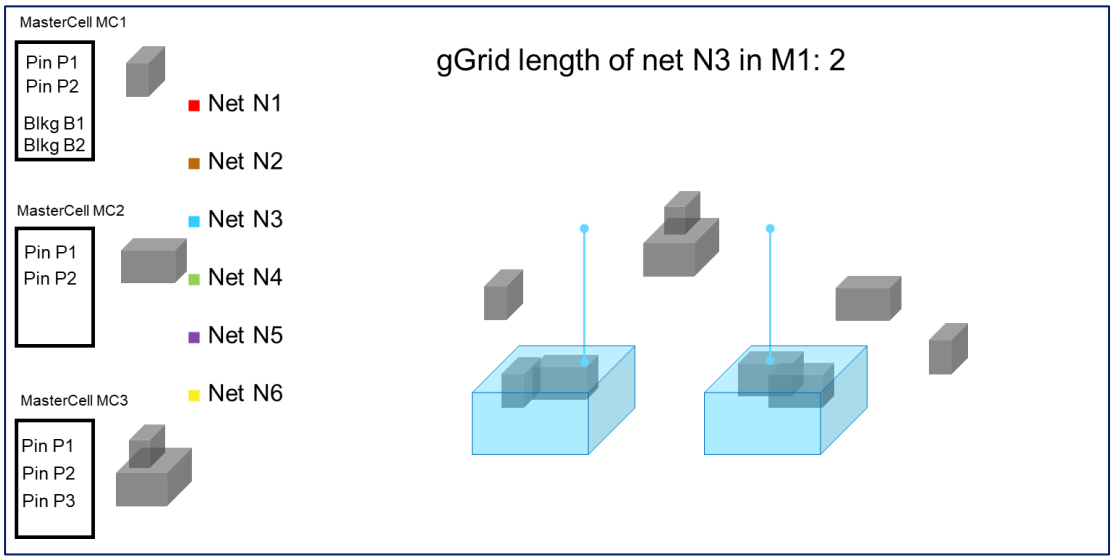

MasterCell MC1
Pin P1
Pin P2
Blkg B1
Blkg B2
MasterCell MC2
Pin P1
Pin P2
MasterCell MC3
Pin P1
Pin P2
Pin P3
Net N1
Net N2
Net N3
Net N4
Net N5
Net N6

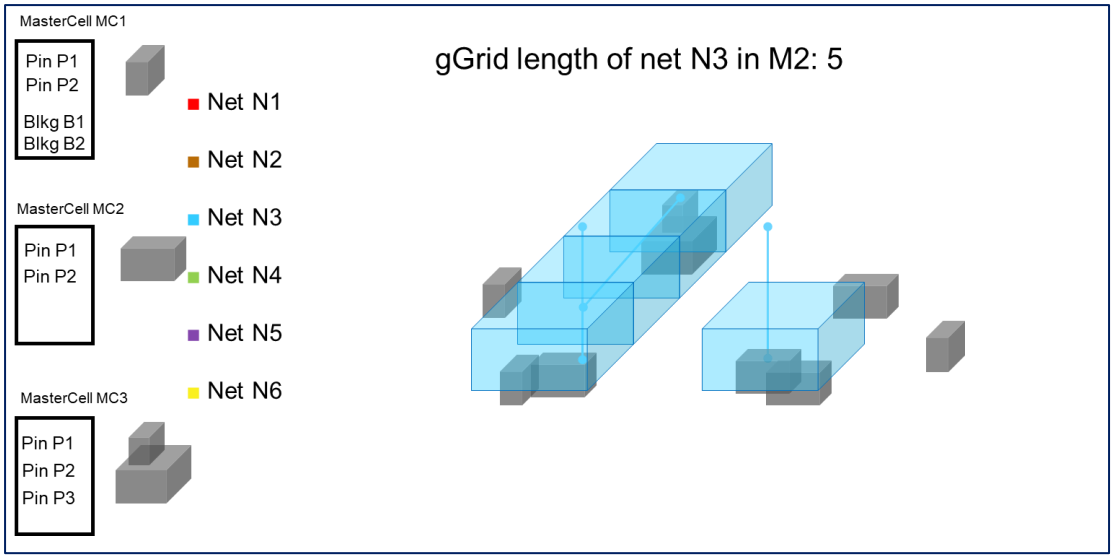

MasterCell MC1
Pin P1
Pin P2
Blkg B1
Blkg B2
MasterCell MC2
Pin P1
Pin P2
MasterCell MC3
Pin P1
Pin P2
Pin P3
Net N1
Net N2
Net N3
Net N4
Net N5
Net N6
gGrid length of net N1 in M1: 3

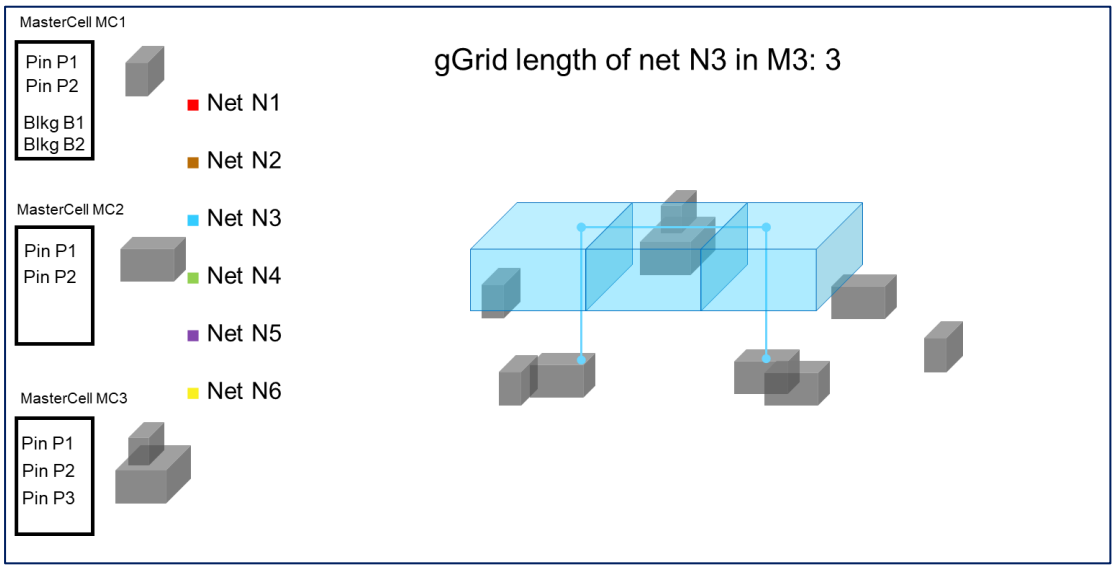

MasterCell MC1
Pin P1
Pin P2
Blkg B1
Blkg B2
MasterCell MC2
Pin P1
Pin P2
MasterCell MC3
Pin P1
Pin P2
Pin P3
Net N1
Net N2
Net N3
Net N4
Net N5
Net N6
gGrid length of net N1 in M2: 8


MasterCell MC1
Pin P1
Pin P2
Blkg B1
Blkg B2
MasterCell MC2
Pin P1
Pin P2
MasterCell MC3
Pin P1
Pin P2
Pin P3
Net N1
Net N2
Net N3
Net N4
Net N5
Net N6
gGrid length of net N1 in M3: 6


MasterCell MC1
Pin P1
Pin P2
Blkg B1
Blkg B2
MasterCell MC2
Pin P1
Pin P2
MasterCell MC3
Pin P1
Pin P2
Pin P3
Net N1
Net N2
Net N3
Net N4
Net N5
Net N6


MasterCell MC1
Pin P1
Pin P2
Blkg B1
Blkg B2
MasterCell MC2
Pin P1
Pin P2
MasterCell MC3
Pin P1
Pin P2
Pin P3
Net N1
Net N2
Net N3
Net N4
Net N5
Net N6
gGrid length of net N3 in M1: 2





5. Evaluation

- No overflow gGrid allowed.
- No open net allowed – All pins of a net must be in the same connected component by using output routes.
- Voltage area constraints must be satisfied.
- Max cell movement constraint must be satisfied
- Runtime limit is 1hr for each case in the evaluation machine. The hidden cases will be in the same scale as public cases.

If the program and the output data violate any of these above bullets, you will get 0 score for the corresponding test case.

- Net min layer constraint & routing direction must be satisfied
    - The min layer constraint limits that the H-dir and V-dir routing must be on

or above the given min layer.

- The routing direction limits that H-dir routings must be on odd layers, V-dir routings must be on even layers.

The violated routing segment(s) of this bullet will be discarded during the evaluation. Be aware that if this causes net open, you will still get 0 score for the corresponding test case.

Evaluation score =

$$\sum_{i=1}^{\#nets} \left(\sum_{j=1}^{\#layers} \left(net_i \text{ gGrid length on } layer_j\right) \times \text{PowerFactor}(layer_j) \right) \times \text{Weight}(net_i)$$

The ranking of this contest is based on the summation of the score of each case including both public and hidden cases.

5.1 Program requirements

Your program should name cell_move_router and we can execute like following:

./cell_move_router <input.txt> <output.txt>

The number of CPU cores available for your program is 8 cores in the evaluation.

5.2 Evaluator

There will be an Evaluator provided in the contest website. Contestants can use this provided Evaluator to validate file format of the outputted file and also the correctness of the result.

./evaluator <input.txt> [<output.txt>]

6. References

K.-S. Hu, M.-J. Yang, T.-C. Yu and G.-C. Chen, "ICCAD-2020 CAD contest in Routing with Cell Movement : Invited Talk," 2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD), San Diego, CA, USA, 2020, pp. 1-4.

X. He, T. Huang, L. Xiao, H. Tian, G. Cui and E. F. Y. Young, "Ripple: An effective routability-driven placer by iterative cell movement," 2011 IEEE/ACM International

Conference on Computer-Aided Design (ICCAD), San Jose, CA, 2011, pp. 74-79.

Yue Xu, Yanheng Zhang and Chris Chu, "FastRoute 4.0: Global router with efficient via minimization," 2009 Asia and South Pacific Design Automation Conference, Yokohama, 2009, pp. 576-581.

W. Liu, W. Kao, Y. Li and K. Chao, "NCTU-GR 2.0: Multithreaded Collision-Aware Global Routing With Bounded-Length Maze Routing," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 32, no. 5, pp. 709-722, May 2013.