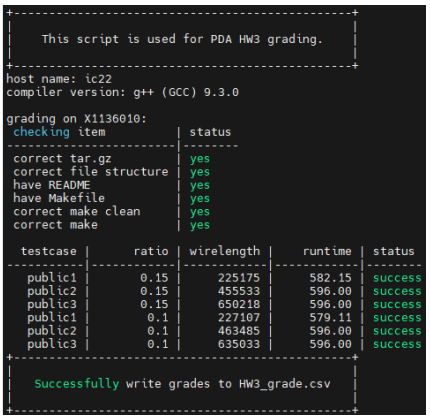
黃偉祥 X1136010

How to

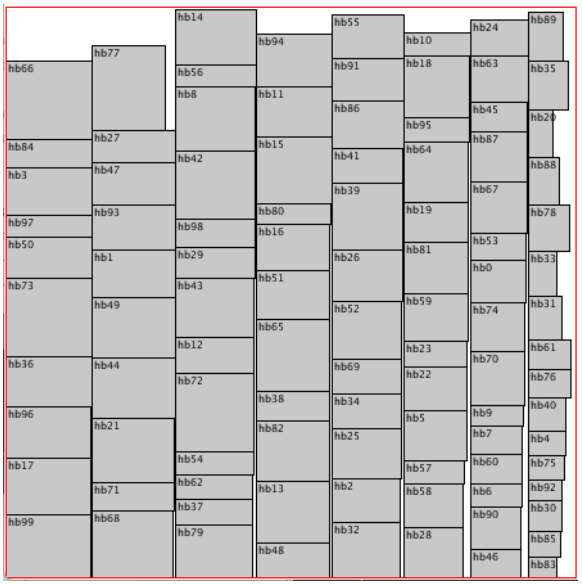
- Under `src/` directory use `make` to compile and command below to execute
  - `binary(hw3) input.txt output.out dead_space_ratio [hpwl_contribution] [penalty_contribution]`
  - ex : `../bin/hw3 ../testcase/public1.txt ../output/public1.out 0.1`
  - `../bin/hw3 ../testcase/public1.txt ../output/public1.out 0.1 0.5`
    - specify **HPWL** contribute **HPWL * 0.5** to cost
  - `../bin/hw3 ../testcase/public1.txt ../output/public1.out 0.1 1 100`
    - specify **HPWL** contribute **HPWL * 1** to cost and **penalty** contribute **penalty * 100** to cost
- Under `src/` directory use `make grade` to grade the work

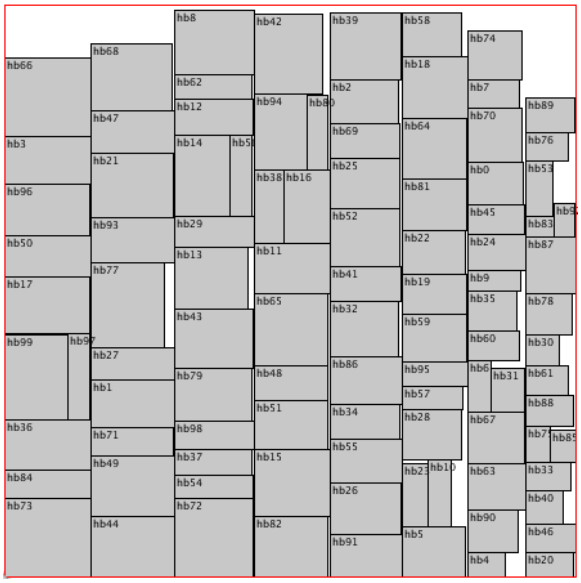
Result



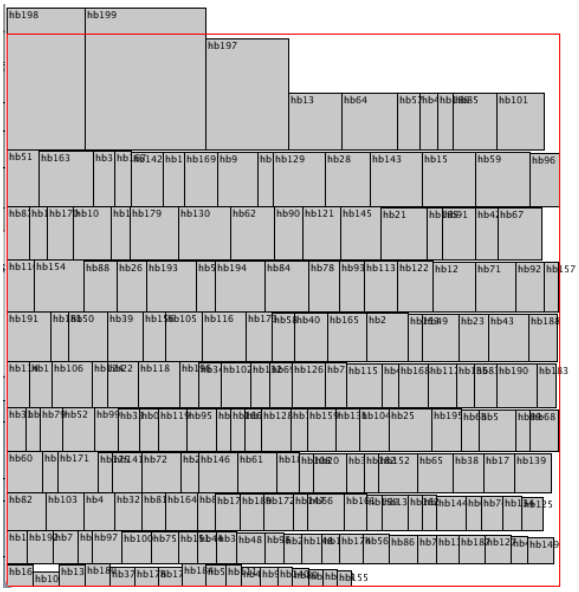
	Wirelength(HPWL)	Wirelength(HPWL)
Case name	Dead ratio = 0.15	Dead ratio = 0.1
public1	225175	227107
public2	455533	463485
public3	650218	635033

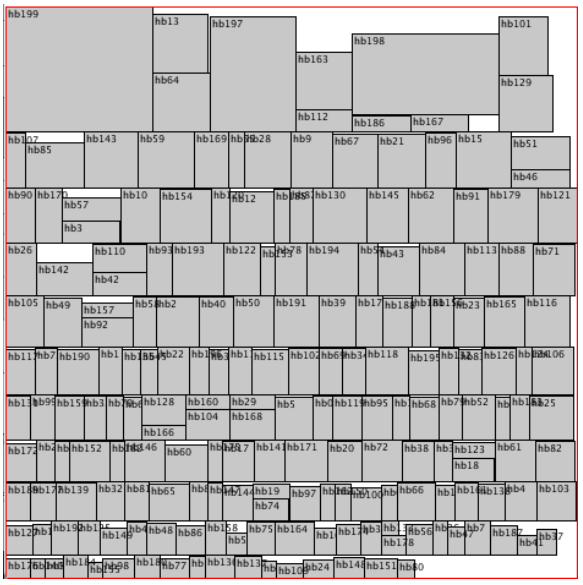
Public1.txt (Initial(left) → Result (right))



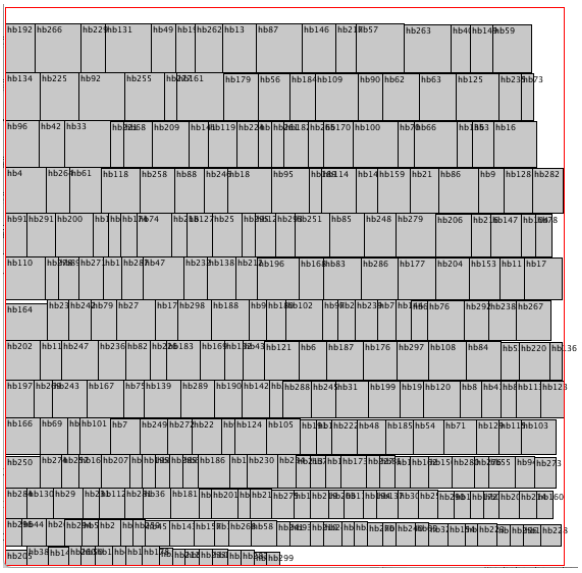


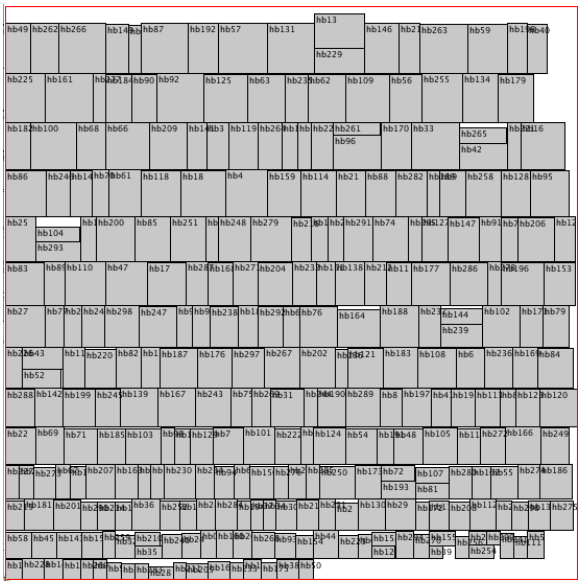
Public2.txt (Initial(left) → Result (right))





Public3.txt (Initial(left) → Result (right))





Experiments

Normal assignment of node (left then right)

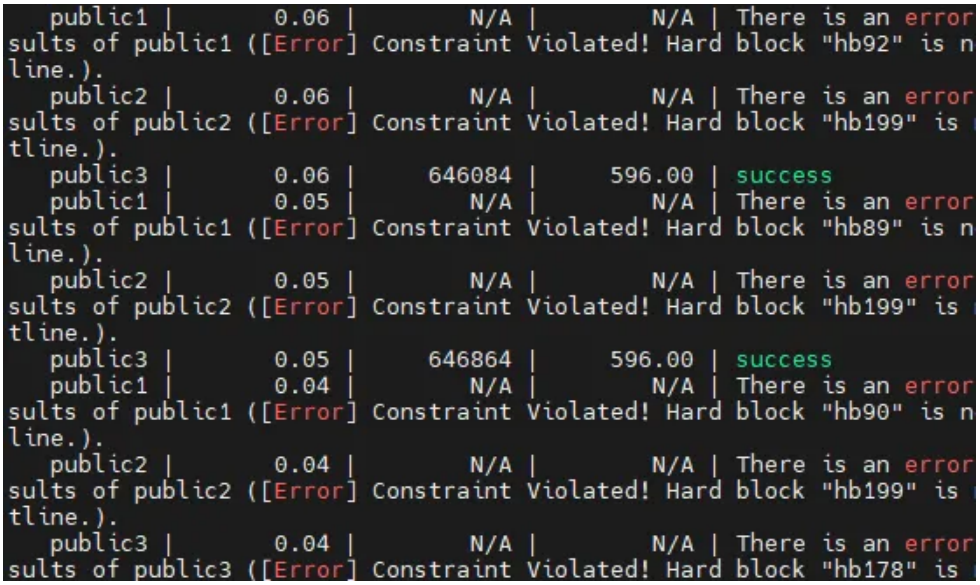
```
  testcase |       ratio | wirelength |    runtime | status
-----------|-------------|------------|------------|--------
   public1 |        0.09 |     231046 |     596.00 | success
   public2 |        0.09 |        N/A |        N/A | There is an error in the output re
sults of public2 ([Error] Constraint Violated! Hard block "hb137" is not within the ou
tline.).
   public3 |        0.09 |     667348 |     596.00 | success
   public1 |        0.08 |     230933 |     593.35 | success
   public2 |        0.08 |        N/A |        N/A | There is an error in the output re
sults of public2 ([Error] Constraint Violated! Hard block "hb178" is not within the ou
tline.).
   public3 |        0.08 |     664731 |     596.00 | success
   public1 |        0.07 |     235606 |     553.79 | success
   public2 |        0.07 |        N/A |        N/A | There is an error in the output re
sults of public2 ([Error] Constraint Violated! Hard block "hb187" is not within the ou
tline.).
   public3 |        0.07 |     651981 |     596.00 | success
   public1 |        0.06 |     252806 |     596.00 | success
   public2 |        0.06 |        N/A |        N/A | There is an error in the output re
sults of public2 ([Error] Constraint Violated! Hard block "hb187" is not within the ou
tline.).
   public3 |        0.06 |     665422 |     596.00 | success
```

```
  testcase |      ratio | wirelength |    runtime | status
-----------|------------|------------|------------|--------
   public1 |       0.05 |        N/A |        N/A | There is
line.).
   public2 |       0.05 |        N/A |        N/A | There is
tline.).
   public3 |       0.05 |     654873 |     596.00 | success
   public1 |       0.04 |        N/A |        N/A | There is
line.).
   public2 |       0.04 |        N/A |        N/A | There is
tline.).
   public3 |       0.04 |     809379 |     596.00 | success
```

	public1.txt (hpwl)	public2.txt (hpwl)	public3.txt (hpwl)
1.0	228159	481540	669264
0.09	231046	failed(hb137)	667348
0.08	230933	failed(hb178)	664731
0.07	235606	failed(hb187)	651981
0.06	252806	failed(hb187)	665422
0.05	failed(hb92)	failed(hb184)	654873
0.04	failed(hb92)	failed(hb180)	809379
0.03	failed(hb89)	failed(hb187)	failed(hb294)

Another way of assigning node, if “H” then right node first

```
  testcase |      ratio | wirelength |    runtime | status
-----------|------------|------------|------------|--------
   public1 |       0.09 |     228514 |     588.01 | success
   public2 |       0.09 |        N/A |        N/A | There is an error
sults of public2 ([Error] Constraint Violated! Hard block "hb199" is
tline.).
   public3 |       0.09 |     649193 |     596.00 | success
   public1 |       0.08 |     227280 |     561.93 | success
   public2 |       0.08 |        N/A |        N/A | There is an error
sults of public2 ([Error] Constraint Violated! Hard block "hb199" is
tline.).
   public3 |       0.08 |     636805 |     596.00 | success
   public1 |       0.07 |     226742 |     570.18 | success
   public2 |       0.07 |        N/A |        N/A | There is an error
sults of public2 ([Error] Constraint Violated! Hard block "hb199" is
tline.).
   public3 |       0.07 |     644245 |     596.00 | success
```



	public1.txt (hpwl)	public2.txt (hpwl)	public3.txt (hpwl)
1.0	227107	463485	635033
0.09	228514	failed(hb199)	649193
0.08	227280	failed(hb199)	636805
0.07	226742	failed(hb199)	644245
0.06	failed(hb92)	failed(hb199)	646084
0.05	failed(hb89)	failed(hb199)	646864
0.04	failed(hb90)	failed(hb199)	failed(hb178)
0.03	failed(hb90)	failed(hb199)	failed(hb277)

Details of implementation

- Using stockmeyer to calculate the compacted area, and M1,M2,M3 moves same as DAC-86 paper

Penalty

- Use penalty to give large cost of the area that excess the fixed-outline
- **penalty = used area *Sum (pow(x_i,2) +pow(y_i,2))**
  - used area = total used width * total used height
  - x_i = width excess fixed-outline of each hardblock
  - y_i = height excess fixed-outline of each hardblock
- We need penalty to let the program know if excess the fixed-outline then it is illegal, if not the program will not focus on the fixed-outline constraint

Temperature Method 1

- When get a valid solution (in fixed-outline), then use 2 rounds to get a new T0
    - Since I give a large penalty to cost function, so if using the same T0 with initial temperature may lead to accept all bad move

```
if(delta_cost > 0 && delta_cost<5000 && (first || second))
   total_delta_cost += delta_cost;num_d_cost++;
// For collect T0, only collect delta_cost < 5000 (penalty too large)
// , collect 2 round

if(!start){start=true;first=true;second=true;T=2000;}
// When get a valid solution(in fexed-outline), then start collecting
// initial T0 for reduce wirelength

if(!first && second){
  second=false;
  T = (total_delta_cost/num_d_cost) / log(P);
}
// After 2 rounds, then calculate new T0 for reduce wirelength

if(first)first=false;
// First round done
```

Parameters (public2.txt, change T when get a valid solution)	Wirelength(HPWL)
T=10000000.0,P=80,epsilon=0.0001,cooling_rate=0.99, k=20	514909
T=10000000.0,P=10,epsilon=0.0001,cooling_rate=0.988, k=20	506893
T=10000000.0,P=5,epsilon=0.0001,cooling_rate=0.99, k=20	501089
T=10000000.0,P=20,epsilon=0.0001,cooling_rate=0.99, k=20	501608
T=10000000.0,P=5,epsilon=0.0001,cooling_rate=0.99, k=30	489021
T=10000000.0,P=2,epsilon=0.0001,cooling_rate=0.99, k=30	489040
T=10000000.0,P=2.5,epsilon=0.0001,cooling_rate=0.99, k=35	failed
T=10000000.0,P=3,epsilon=0.0001,cooling_rate=0.985, k=30	508520
T=10000.0,P=3,epsilon=0.0001,cooling_rate=0.985, k=30	failed
T=10000.0,P=5,epsilon=0.0001,cooling_rate=0.99, k=30	493924

Temperature Method 2

- Add calculate T0 for initial Temperature (for fixed-outline), method 1 use as usual
    - Same as DAC-86 paper, use delta_cost to accumulate for initial temperature T0

```
if(delta_cost > 0)tmp+=delta_cost;
// Collect delta_cost for initial Temperature

if(count==N*10 && !start)T = tmp/(count*fpRegion*fpRegion*10);
// Collect for N*10 iterations,
// if before iteration get a valid solution,
// then will not be used
```

Parameters (public2.txt, method 2 to change T)	Wirelength(HPWL)
T=100.0,P=5,epsilon=0.0001,cooling_rate= 0.92, k=30	498978
T=100.0,P=5,epsilon=0.0001,cooling_rate= 0.97, k=30	485940
T=100.0,P=10, epsilon=0.0001,cooling_rate= 0.97, k=30	486593
T=100.0,P=10, epsilon=0.0001,cooling_rate=0.9, k=15, T=2000	483692
T=100.0,P=1.2 epsilon=0.0001,cooling_rate=0.9, k=15, T=2000	482416
T=100.0,P=1.7,epsilon=0.0001,cooling_rate=0.95, k=20, T=2000	489845
T=100.0,P=2,epsilon=0.0001,cooling_rate=0.97, k=25, T=2000	487027
T=100.0,P=2, epsilon=0.0001,cooling_rate=0.97, k=15, T=2000	481530

Initial floorplan method

```
vector<string> initialFP(){
    auto [best_pe1, best_cost1] = initialbyHeight();
    auto [best_pe2, best_cost2] = initialbyWidth();
    for(int i=0;i<1000;i++){
        auto [pe1, cost1] = initialbyHeight();
        auto [pe2, cost2] = initialbyWidth();
        if(cost1 < best_cost1){
            best_pe1 = pe1;
            best_cost1 = cost1;
        }
        if(cost2 < best_cost2){
            best_pe2 = pe2;
            best_cost2 = cost2;
        }
    }
    vector<pair<vector<string>, long long int>> plans = {
        {best_pe1, best_cost1},
        {best_pe2, best_cost2}
    };
```

```
    auto best_plan = plans[0];
    for (const auto& plan : plans)if (plan.second < best_plan.second)
        best_plan = plan;

    return best_plan.first;
}
```

- InitialbyHeight()
  - shuffle and sort by height (rotate if width > height)
  - start from bottom left put hard block, from left to right
    - if excess fixed-outline width then put in next row
- InitialbyWidth()
  - shuffle sort by width (rotate if height> width)
  - start from bottom left put hard block, from bottom to top
    - if excess fixed-outline height then put in next column
- Do 1000 iteration to get a lowest cost of initial
  - May have same height/width so shuffle may give difference initial solution(HPWL, area)

Tricks

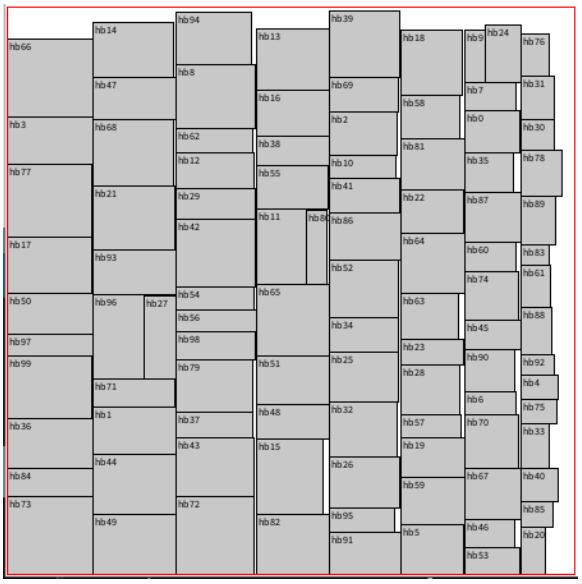
1. Large penalty cost to let program get a valid solution first
2. Two different initial solution for valid and invalid
   a. Before get a valid solution we have large penalty cost so need a larger initial temperature (if not may unable to get a valid solution, because it will not accept all bad move and may stuck at local minimal)
   b. After get a valid solution we can have a smaller temperature to prevent always accept bad move (if use same temperature with invalid solution)
3. Shuffle to get a better initial solution, better solution lead to get a better result and get a valid solution faster
4. Another way of **assigning** blocks, assign **right** node first if "**H**"
   a. Since most of the pins are at bottom, and most small blocks are connected with pin, so we can put right node first to make the small blocks close to bottom

Public1.txt

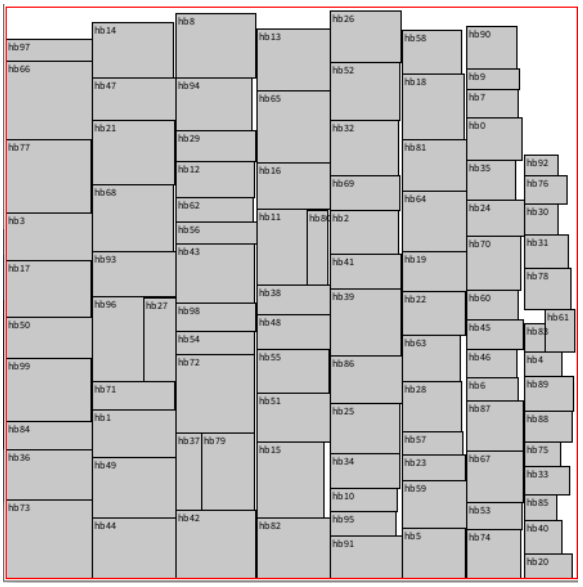
- 280806 (230015) —> 280806 (233168) —> 274065(228159) —> 272707(227790)

- No penalty result → with penalty no shuffle → with penalty with shuffle —> Assign right node first if “H”
- initial cost (final cost)

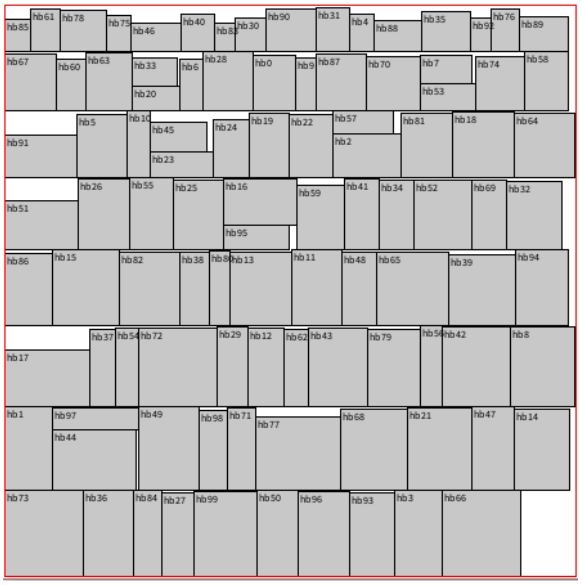
- Without penalty



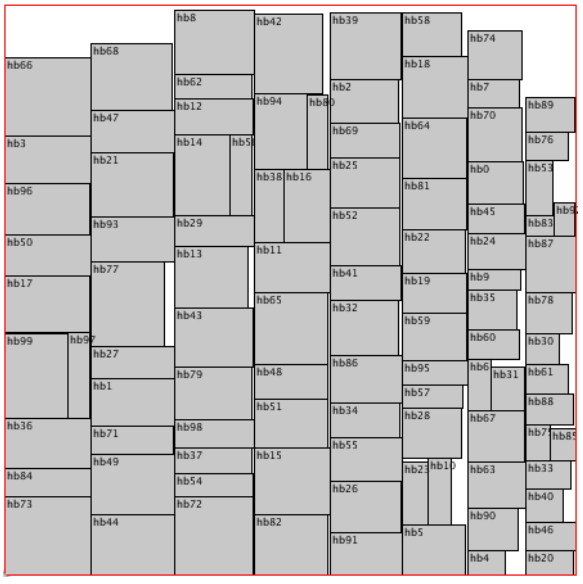
- With penalty



- With shuffle



- Assign right node first if “H”



- In public1.txt we get a valid solution at initial floor planning, so using penalty may not necessary get better solution.

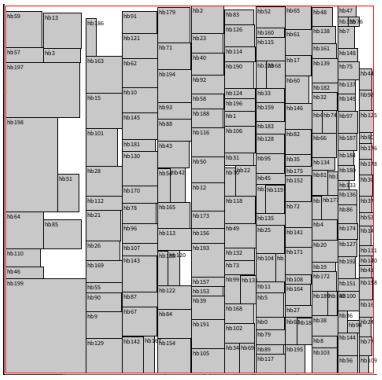
Public2.txt

- 799331 (701693) —> 411761503921 (559323) —> 411761503921 (492308) —> 253689039493 (481530) —> 38685250198 (464833)
  - No penalty result —> Add penalty to cost function —> Use 2 initial temperature methods —> Add shuffle at initial floor plan —> Assign right node first if “H”
  - initial cost (final cost)

- Without penalty

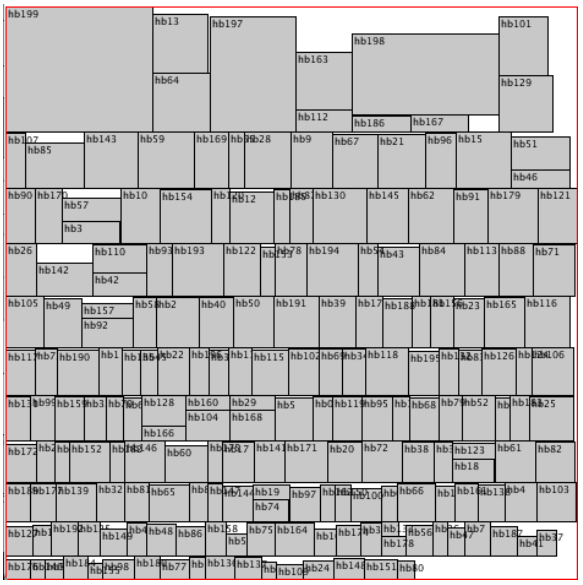
- Add penalty

- Add 2 initial methods



- Add shuffle at initial floor plan

- Assign right node first if “H”



- It will get an invalid solution at the end if not add penalty to it, and using 2 initial temperature and shuffle tricks will improve a lot for the final result
- Different assignment order can lead to better result due to more small blocks has connected to net in this case

Conclusion

1. Important of a better initial solution
    a. A better initial solution may always lead to a better final solution
2. Use a large penalty and wirelength as cost function can solve fixed-outline floor planning
    a. But after get a valid solution may need to recomputed another initial temperature to it
3. After **stockmeyer** algorithm to calculate compacted area, it is hard to calculate the (x,y) coordinates for all hard blocks
    a. AI is a good tools to have a idea to start
4. Different assignment order may lead to better result