A New Clustering Approach and Its Application To BBL Placement


* Software Laboratory, Academia Sinica
** Tsinghua University, Beijing
*** Peking University, Beijing

ABSTRACT

A new approach for clustering applied to Building Block placement is presented. Unlike traditional approaches, which only consider local factors such as connectivity and shape matching of blocks in a cluster, our approach (called the GAC method) not only considers connectivity and shape matching of clusters, but also takes a more important factor, global analyses, into account. Furthermore, in this paper we have proposed a new shape matching and a propagating method to compute the connectivity of clusters. The new method makes connectivity computation more precisely. This clustering algorithm has been implemented in Fortran 77 on Micro VAX II. The experimental placement results are much better than BEAR and TimberWolfMC, the latter of which adopts the simulated annealing approach, and the GAC method has a much better run time.

1. Introduction

With the rapid development of IC technology, the Building Block Layout becomes a more and more important approach for VLSI physical design. Because of the complexity of BBL, the placement has become a key technique in design efficiency and effectiveness of the chip layout. It is well known that slowness in placement and difficulty in improvement are characteristics of BBL placement. So high quality and high speed are the objects for BBL placement. The placement algorithm based on clustering approach is a good method because the global routing and routing area allocation can be made simultaneously in the placement procedure so as to get a good and consistent placement result. In this method, placement result strongly depends on the clustering result. Unfortunately, all traditional clustering methods [1 to 5] only consider local factors such as connectivity and shape matching of blocks in a cluster. It is difficult to achieve global optimization. Our global analysis clustering (GAC) method not only considers connectivity and shape matching but also is based on global analyses, which include the clustering tree depth, the relationship of node degrees of every level on the clustering tree, and the matching between clustering approach and placement algorithm.

We discuss the number of sub-clusters of a cluster in Section 2, the relationship between clustering tree depth and placement area utilization in Section 3, the relationship of node degrees at every level in the clustering tree in Section 4, the matching problem between clustering methodology and placement algorithm in Section 5, the connectivity of clusters in Section 6, Section 7 is about block shape matching. Section 8 describes our GAC algorithm in detail. Some experimental placement results applying the GAC method are presented and comparisons with BEAR and TimberWolfMC results are given in Section 9. The last section is the conclusion.

2. Number of Sub-clusters of a Cluster

For a cluster with Nc sub-clusters, the number of distinct templates is given in paper [4] (see Table 1).

A template [3] has Nc rooms if its corresponding cluster has Nc sub-clusters. For one template there are Nc room assignments, and for one assignment if the pin direction is considered, there are 2**n area orientation combinations. Table 1 gives details.

Table 1 tells us that the higher the number of sub-clusters of a cluster, the higher the number of combinations. From statistical law, we know that the more the combinations, the greater the possibility to get a better result. Table 2 lists the average area usages of the clustering approach in which the corresponding cluster has various sub-cluster numbers. In view of time complexity, the number of sub-clusters is limited to 5.

It is obvious that the area usage will be 100% if the number of sub-clusters is 1.

Table 2 shows that the number of sub-clusters of a cluster should be as large as possible if you want to obtain a high area usage clustering result.

3. Relationship of Clustering Tree Depth and Placement Area Utilization

Because the number of blocks of a circuit is fixed, it is easy to find that the smaller the clustering tree depth, the higher the average number of sub-clusters of the clusters. We have carried on many experiments. Table 3 is the list of bottom-up placement results with various clustering tree depths. In this circuit, the number of blocks is 33, the number of nets is 159. Similar results have been obtained for other circuits.

The above analysis shows that a clustering tree depth should be as small as possible if you want to get a good placement result. This is the first conclusion of our global analyses.

In the following, we will compute the tree depth.

The tree depth should be as small as possible according to the first conclusion. Let n be the number of blocks of a circuit, m the maximum number of children of a cluster, D the tree depth. Then, the minimum tree depth is

\[ D_{\text{min}} = \text{ceiling} \left( \frac{\ln(n)}{\ln(m)} \right) \]

4. Relationship of Node Degrees at Every Level in the Clustering Tree

In traditional clustering methodology, the number of sub-clusters of a cluster (or the degree of a node corresponding to the cluster) is determined in terms of the connectivity and shape matching among sub-clusters.

In fact, the placement area usage relates to the degrees of nodes on
different levels of the clustering tree. At low levels, the areas of clusters are relatively small and at high levels relatively large, especially at the root of the clustering tree. If the usage rates of placement area of clusters at different levels are the same, the dead area of the clusters at a higher level will be much more than the one of the clusters at a lower level. Furthermore, the area usage mainly depends on the degree of the node as shown in section 2. Therefore, if you want to get a placement with high area usage you may group the clusters according to the following conclusion: the degrees of nodes at high levels should be the maximum or 1. The results listed in Table 4 support this conclusion. This is the second conclusion of our global analyses.

The second conclusion tells us that the degrees of nodes at high levels should be equal to the maximum or 1. To get a good placement result, such levels that are higher than or equal to the second level (suppose the leaf level is the zeroth level) are defined as the high levels. The degrees will be determined according to the minimum tree depth and by giving considerations of connectivity and shape matching.

5. Matching Between Clustering Approach and Placement Algorithm

Nowadays, placement algorithms [3, 5] based on clustering always adopt the following strategy: do bottom-up clustering and then top-down placement including global routing and routing area allocation. The bottom-up clustering has negligible CPU time as compared to top-down placement. The top-down placement places the clusters from the root to the leaf level according to the clustering tree. Time complexity for placement at a non-leaf level is greatly different from the one at a leaf level because the pin directions and locations will not be considered at non-leaf level but they must be taken into account at the leaf level. Therefore, time complexity at the leaf level equals that at a non-leaf level. *F*^2*Nc*, where *Nc* is number of blocks of a cluster (generally *Nc* <= 4). Because high speed is one of the objects of BBL placement, it is necessary to develop a clustering algorithm which will result in a placement with high speed by matching the clustering algorithm and placement algorithm.

From global analyses and the consideration of time complexity of top-down placement mentioned above, it is obvious that a clustering tree whose node degrees at the first level are small will result in a fast placement. But Table 2 shows that the area usage of a cluster with two children is very low in general. To get a satisfactory placement result, the speed and quality (area usage) must be taken into consideration. The conclusion of this placement/clustering matching analysis is: In general, the node degree of the first level should be 1 (100% area usage), 4 or 3 (in the case that the maximum number of sub-clusters <= 4), and 2 is selected as seldom as possible.

In addition, the degrees will also be determined according to the minimum tree depth and by giving considerations to connectivity and shape matching.

6. Connectivity of Clusters

In this paper, a pairwise clustering [7] method is adopted. By introducing a new propagation method to compute connectivity, our pairwise clustering is very effective.

Let *i* and *j* be sub-clusters, *C*^*ij* in the sum of weights of edges (or connections) between *i* and *j*. Let *P*(*i, j*) be the cluster to which *i* and *j* belong. After subcluster *i* and *j* are grouped into a cluster, *P*(*i, j*) becomes a cluster *k* and can group with other clusters again as long as the number of its children is less than the maximum number. The clusters grouped into a cluster *k* at one level are called the children of the cluster *k*. The number *I*^*i* of inner connections of sub-cluster *i* is

\[ I^i = C^*_ij \]

and *I*^*j* is

\[ I^j = C^*_kj = C^*_ij \]

Let *T*^*i* be the sum of connections connected to cluster *i*. Then, the number of outer connections of cluster *i* is

\[ E^i = T^i - C^*_ij \]

Generally, the criterion to evaluate if sub-clusters *i* and *j* should be grouped into cluster *k* is:

\[ F = \frac{I^i}{E^i} + \frac{I^j}{E^j} \]

This criterion may seem unreasonable for some cases. Let's see Fig. 1.

![Fig. 1 Connectivity among clusters](image)

From Fig. 1, we can obtain *F*^13 = *F*^12 = *F*^23 = 2 (suppose all weights are 1), which will lead to an unreasonable result that sub-cluster 1, 2 and 3 cannot be grouped into one cluster in the pairwise clustering procedure. To overcome this serious shortcoming, we present a new method, the propagation method to compute *F* between cluster *i* and cluster *j*.

The propagation method is described as follows:

In discussing the *F* value between cluster *i* and cluster *j*, the number of inner connections of cluster *i* not only relates to *C*^*_ij*, but also relates to the indirect connections propagated by other clusters. In view of time complexity, the indirect connections propagated by one cluster which does not belong to *P*(*i, j*) will be taken into account. The computation of *I* is redefined as follows:

\[ I^i = C^*_ij + \sum_{k \neq i} \min(C^*_ik, C^*_kj)/2 \]

The index *k* runs over all clusters created at present except for cluster *i* and cluster *j*. The factor 2 comes from the factor (1 + number of clusters propagating indirect connections) when the number is 1.

The number of outer connections of cluster *i* can be similarly redefined:

\[ E^i = T^i - I^i \]

Thus, the propagation method is introduced. This method can be used in various clustering approaches. The connectivity of Fig. 1 can be computed as follows:

\[ F^13 = F^12 = F^23 = (2+1)/1 + (2+1)/1 = 6 \]

This value is high and these three clusters can be grouped into a cluster in general.

7. Shape Matching

Clustering based only on connectivity information can result in a block shape mismatching that makes it impossible for the placement algorithm to avoid large dead space areas.

It is obvious that two clusters or blocks do not match if their areas or the lengths of their longer sides are sufficiently different. Shape matching is considered in three steps.
1. Define criterion 1 that a block can not be clustered until level (L+1).

Criterion 1: the blocks whose \( B > \sqrt[3]{(2^n+1)} \) will not be clustered until level \( L+1 \) under the constraint that the minimum tree depth must be satisfied. If more than one block satisfies the constraint of \( B \) and the constraint of minimum tree depth, the block with maximum \( B \) (and maximum area if more than one such block has maximum \( B \)) is selected. All blocks must be clustered at the root level.

Here \( B \) is the longer side of a block, \( S \) is the sum of areas of all blocks, \( NL \) is the number of clusters in the \( L \)th level. The constraint \( B > \sqrt[3]{(2^n+1)} \) means that the block whose long side is longer than the side of a square whose area is equal to half of the average area of all clusters in level \( L \).

2. Consider the leaf level particularly.

Let \( nL \) be the number of nodes at level 1 less the number of blocks which should be grouped at the levels higher than 0, \( mL \) the number of blocks which should be clustered at the leaf level. If \( mL \) is less than or equal to \( 4*nL-3 \), \( nL \) should be decreased until \( mL > 4*nL-3 \). The method stated above, which is called LeafParticularDeal, is described as follows.

\[ \text{LeafParticularDeal}(\cdot) \]
\[ \begin{align*}
& \text{do} \\
& \begin{cases}
& \text{Compute connectivity matrix } C_{ij} \quad (i=1, mL; j=1, mL); \\
& \text{for } (i=1; i < mL + 1; i++) \\
& \quad \{ \\
& \quad \quad C_{ij} = \max(C_{ij}, j=1, mL); \\
& \quad \quad \} \\
& \quad \} \\
& \text{Select the block } k \text{ with the minimum } C_{ij}; \\
& \quad \text{if there are two or more blocks with the minimum } C_{ij}, \\
& \quad \quad * \text{ the block with the minimum sum of } C_{ij} (j=1, mL) \\
& \quad \quad \text{is selected. If the sum is equal, arbitrary one is selected} \\
& \quad \text{}/ \\
& \quad \text{Delete block } k \text{ in the matrix}; \\
& \quad mL = mL - 1; \\
& \quad \text{let block } k \text{ enter into a queue in which all blocks} \\
& \quad \text{should be grouped at level } 1; \\
& \quad \} \\
& \text{while (} mL \leq 4*nL - 3; \} \\
\end{align*} \]

3. Define a shape matching factor \( f \).

Level 0 (or the leaf level):

Level 0 is particularly defined because the dimensions of nodes at level 0 must be considered. Let \( B_i \) be the longer side of block \( i \), \( M_i = B_i - B_j, Nc \) is the number of children of the cluster to which blocks \( i \) and \( j \) belong, \( S_i \) is the area of block \( i \) or cluster \( i \),

\[ SM_{ij} = S_i - S_j, \]

\[ f = \sqrt{\sum_{i,j} (M_i^2) / \sum_{i} (M_i)^2} + \sqrt{\sum_{i,j} (SM_{ij})^2 / \sum_{i,j} S_i} \]

Nonleaf level:

\[ f = \sqrt{\sum_{i,j} (SM_{ij})^2 / \sum_{i,j} S_i} \]

In pairwise clustering, we must merge the connectivity and shape matching factors together to evaluate a cluster. The objective function is defined as follows:

\[ V = F - \alpha \times f \quad (0 \leq \alpha \leq 5) \]

The value of \( \alpha \) is adjustable to obtain a satisfactory clustering. If \( \alpha \) is equal to 0, it means that shape matching is ignored.

In a word, a good clustering methodology must take and precisely take into accounts the tree depth, the relationship of node degrees at different levels, the matching with the placement algorithm, the connectivity of blocks, and the matching of block shapes.

8. The GAC Algorithm

(1) Basic Idea

The GAC clustering algorithm is divided into top-down and bottom-up procedures. In the top-down procedure, the tree depth is computed, the blocks which should be grouped at some level are selected and the number of clusters at every level is determined according to our global analyses and the first step of shape matching mentioned in Section 7. The bottom-up procedure creates the clustering tree according to the second and third steps and the objective function \( V \).

(2) Brief Description of GAC Algorithm

GlobalAnalyseClustering()

\[ */ \text{ top-down procedure } */ \]

compute tree depth \( D_{min} \);

for (Nlevel = \( D_{min}+1 \); Nlevel \( > = 0 \); Nlevel \( -- \))

\[ \begin{cases}
& \text{ selecting the blocks which should be grouped on level } Nlevel+1; \\
& \text{ compute the number of clusters at level } Nlevel; \\
& \text{ call LeafParticularDeal();} \\
\end{cases} \]

\[ \text*/ \text{ bottom-up procedure } */ \]

for (Nlevel = 1; Nlevel \( <= D_{min} \); Nlevel \( ++ \))

\[ \begin{cases}
& \text{ } C_n \text{ number of clusters at level } Nlevel; \\
& \text{ } */ \text{ select } C_n \text{ seeds from the clusters and blocks at level } (Nlevel-1) \text{ that should be grouped at } Nlevel; \\
& \text{ } */ \text{ select first seed from clusters at level } (Nlevel-1) \text{ arbitrarily; } \\
& \text{ SeedNum } = 1; \text{ } */ \text{ number of seeds } = 1 */ \text{ while } (\text{SeedNum } < C_n)
\end{cases} \]

\[ \begin{cases}
& \text{ for } (i = 0; i < \text{number of non-seeds at present}; i++) \\
& \quad \text{ for } (j = 0; j < \text{SeedNum}; j++) \\
& \quad \quad \text{ compute } V_k; \\
& \quad \quad \text{ sumVs}(j) += V_k; \\
& \quad \quad \} \\
& \quad \quad \text{ Vk } = \max (\text{ sumVs}(j), \text{SeedNum} - 1); \\
& \quad \quad \text{ select the block/cluster whose } \text{ Vk } = \min (\text{ Vk } \text{ k } = 0, \text{ non-seed number } - 1); \\
& \quad \quad \text{ select the block/cluster as seed } ; \\
& \quad \quad \text{ if there are more than one blocks/cluster whose } \text{ Vk }'s \text{ are minimum, select the one whose sumV is minimum. } \\
& \quad \quad \text{ if sumVs are equal, arbitrarily select one.} \\
& \quad \quad \} \\
& \quad \quad \text{ SeedNum } += 1; \\
& \quad \text{ number of non-seeds } = 1; \\
\end{cases} \]

cluster these seeds in parallel in terms of objective function \( V \), subject to the condition that the maximum number of children is 4; }

9. Experiment Results and Comparisons
In our experiments, the placement system is a subsystem of FRACT, which is a new BBL layout system [9]. Many experiments of placement applying the GAC method and other clustering methods have been done. These clustering methods are GREEDY, RANDOM and MATCHING of the BEAR system [5, 6]. Table 5 is the results and comparisons of four circuits without routing area allocation, and Table 6 has the routing area allocation.

From Table 5 we can find that the average reductions in chip area, in the sum of net lengths (SN) and in CPU time of the four circuits compared with GREEDY are 18%, 0.5% and 45% respectively. The average reductions in the three categories above, compared with RANDOM, are 17.4%, 24% and 62% respectively. The average reductions compared with MATCHING are 11%, 3.1% and 65.4% respectively. The maximum reductions in chip area, sum of net lengths and CPU time are 33%, 39% and 91% respectively. Table 6 also shows that our GAC approach outperforms other clustering techniques.

The comparison to TimberWolfMC [6] is also made. TimberWolfMC is a floor planning tool using simulated annealing. This comparison is not very precise because the methods of computing net length are different. Only circuit I1 and D1 comparisons are available because TimberWolfMC has not been run on circuits iccad and myct. The comparison is listed in Table 7.

10. Conclusion

This paper presents a new clustering (named GAC) approach used for building block placement. Global analyses and the propagation method for computing connectivity and the new shape matching make the GAC method very effective. The placement results based on the GAC method are better than the results obtained in BEAR and TimberWolfMC. Our GAC method can also be used for standard cell, gate array, gate matrix layout by ignoring the constraints of tree depth and node degree.

### Table 1: the number of combinations of a cluster Vs. the number of sub-clusters

<table>
<thead>
<tr>
<th># of sub-clusters</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td># of templates</td>
<td>1</td>
<td>2</td>
<td>6</td>
<td>22</td>
<td>92</td>
</tr>
<tr>
<td># of room assignments</td>
<td>1</td>
<td>2</td>
<td>6</td>
<td>24</td>
<td>120</td>
</tr>
<tr>
<td># of block orientations</td>
<td>8</td>
<td>64</td>
<td>512</td>
<td>4096</td>
<td>32768</td>
</tr>
<tr>
<td># of combinations</td>
<td>8</td>
<td>256</td>
<td>18432</td>
<td>2162688</td>
<td>361778720</td>
</tr>
</tbody>
</table>

### Table 2: clustering results Vs. the number of sub-clusters

<table>
<thead>
<tr>
<th># of sub-clusters</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td># of experiment times</td>
<td>300</td>
<td>300</td>
<td>300</td>
<td>300</td>
</tr>
<tr>
<td>average area usage</td>
<td>81.3%</td>
<td>88.6%</td>
<td>92.4%</td>
<td>96.1%</td>
</tr>
</tbody>
</table>

### Table 3: clustering tree depth Vs. bottom-up placement result

<table>
<thead>
<tr>
<th>clustering tree depth</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td># of experiment times</td>
<td>30</td>
<td>30</td>
<td>30</td>
</tr>
<tr>
<td>average placement area usage</td>
<td>75.7%</td>
<td>62.0%</td>
<td>47.3%</td>
</tr>
</tbody>
</table>

### Table 4: average degree of nodes at different level Vs. placement area usage. (The circuit has 33 blocks and 159 nets)

<table>
<thead>
<tr>
<th>tree depth</th>
<th>first level</th>
<th>second level</th>
<th>root level</th>
<th>times of experiments</th>
<th>average area usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>1.32</td>
<td>5</td>
<td>5</td>
<td>30</td>
<td>88.1%</td>
</tr>
<tr>
<td>3</td>
<td>2.06</td>
<td>4</td>
<td>4</td>
<td>30</td>
<td>79.8%</td>
</tr>
<tr>
<td>3</td>
<td>3.75</td>
<td>4</td>
<td>3</td>
<td>30</td>
<td>74.2%</td>
</tr>
<tr>
<td>3</td>
<td>3.67</td>
<td>3</td>
<td>3</td>
<td>30</td>
<td>70.5%</td>
</tr>
<tr>
<td>3</td>
<td>4.10</td>
<td>4</td>
<td>2</td>
<td>30</td>
<td>67.4%</td>
</tr>
</tbody>
</table>

*The first level means the tree's parent level.*

### Table 5: the comparison of placement results with no routing area allocation based on various clustering methods.

<table>
<thead>
<tr>
<th>circuit</th>
<th>No. cells</th>
<th>No. nets</th>
<th>No. sub-clusters</th>
<th>clustering methods</th>
<th>area SN</th>
<th>area usage</th>
<th>CPU (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>iccad</td>
<td>8 22 34</td>
<td></td>
<td></td>
<td>BEAR greedy</td>
<td>900920</td>
<td>791200</td>
<td>24450</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>random</td>
<td>9009640</td>
<td>756000</td>
<td>26550</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>matching</td>
<td>8609200</td>
<td>791200</td>
<td>24580</td>
</tr>
<tr>
<td>GAC method</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>10409680</td>
<td>707200</td>
<td>25450</td>
</tr>
<tr>
<td>D1</td>
<td>121 38</td>
<td>452</td>
<td></td>
<td>BEAR greedy</td>
<td>1594519</td>
<td>31955</td>
<td>4854</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>random</td>
<td>1084179</td>
<td>30072</td>
<td>8425</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>matching</td>
<td>774167</td>
<td>92559</td>
<td>5166</td>
</tr>
<tr>
<td>GAC method</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1594176</td>
<td>27984</td>
<td>5811</td>
</tr>
<tr>
<td>D2</td>
<td>41 288</td>
<td>837</td>
<td></td>
<td>BEAR greedy</td>
<td>201453</td>
<td>12603</td>
<td>2541</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>random</td>
<td>109972</td>
<td>13680</td>
<td>2579</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>matching</td>
<td>180963</td>
<td>11340</td>
<td>25071</td>
</tr>
<tr>
<td>GAC method</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>771454</td>
<td>29234</td>
<td>20246</td>
</tr>
</tbody>
</table>

### References


Table 6: The comparison of placement results with routing area allocation based on various clustering methods.

<table>
<thead>
<tr>
<th>circuit</th>
<th>No. cells</th>
<th>No. nets</th>
<th>No. pads</th>
<th>No. place</th>
<th>clustering methods</th>
<th>area</th>
<th>SN</th>
<th>CPU (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>iccad</td>
<td>8</td>
<td>22</td>
<td>34</td>
<td>154</td>
<td>BEAR</td>
<td>greedy 4887957 =848859</td>
<td>25341</td>
<td>121</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>random 9179856 =972288</td>
<td>24814</td>
<td>122</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>matching 883956 =846060</td>
<td>25546</td>
<td>125</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>GAC method</td>
<td>1090713 =777170</td>
<td>27077</td>
<td>18</td>
</tr>
<tr>
<td>l1</td>
<td>33</td>
<td>121</td>
<td>38</td>
<td>452</td>
<td>BEAR</td>
<td>greedy 241214 =53518</td>
<td>7111</td>
<td>356</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>random 240212 =50880</td>
<td>11200</td>
<td>459</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>matching 234194 =45396</td>
<td>3978</td>
<td>394</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>GAC method</td>
<td>210200 =63300</td>
<td>7160</td>
<td>206</td>
</tr>
<tr>
<td>dl</td>
<td>17</td>
<td>288</td>
<td>0</td>
<td>837</td>
<td>BEAR</td>
<td>greedy 317203 =65359</td>
<td>45839</td>
<td>260</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>random 279258 =79326</td>
<td>60218</td>
<td>204</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>matching 229304 =66616</td>
<td>44690</td>
<td>204</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>GAC method</td>
<td>229266 =66914</td>
<td>43814</td>
<td>263</td>
</tr>
<tr>
<td>mycct</td>
<td>47</td>
<td>399</td>
<td>48</td>
<td>1171</td>
<td>BEAR</td>
<td>greedy 899806 =771452</td>
<td>222302</td>
<td>751</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>random 899841 =74664</td>
<td>156105</td>
<td>782</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>matching 826936 =690526</td>
<td>148381</td>
<td>635</td>
</tr>
</tbody>
</table>

Table 7: Comparison with TimberWolfMC

<table>
<thead>
<tr>
<th>circuit</th>
<th>result id.</th>
<th>area</th>
<th>SN</th>
<th>CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>l1</td>
<td>our result</td>
<td>210*200</td>
<td>7160</td>
<td>3.5min.</td>
</tr>
<tr>
<td></td>
<td>TimberWolfMC</td>
<td>236*223</td>
<td>7481</td>
<td>45min.</td>
</tr>
<tr>
<td>dl</td>
<td>our result</td>
<td>229*266</td>
<td>43814</td>
<td>4.5min.</td>
</tr>
<tr>
<td></td>
<td>TimberWolfMC</td>
<td>243*305</td>
<td>73565</td>
<td>25min.</td>
</tr>
</tbody>
</table>