Applying the P-Medians in the Design of Modern Systems-on-Chip

Nadezhda Matveeva, Lev Kurbanov, Elena Suvorova
Saint-Petersburg State University of Aerospace Instrumentation
{nadezhda.matveeva, lev.kurbanov}@guap.ru,
suvorova@aanet.ru
Devices for Internet of Things (IoT)

Architecture:
• homogeneous
• heterogeneous

Requirements:
• low energy consumption
• small area
• high speed of data transfer between chip’s nodes and external devices
Development of new chip

It is important to arrange IP-blocks on chip according to requirements such as distance, throughput and etc.

• if system contains several controllers accessing external memory
• if system is to communicate with different external interfaces such as PCIExpress, USB, HDMI and others
• if specialized IP-blocks are included in System-on-Chip (SoC)

The P-medians searching is useful!
## Methods for p-mediands searching

<table>
<thead>
<tr>
<th></th>
<th>An approximate algorithm</th>
<th>Method “traversing a path”</th>
<th>Direct tree search method</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Advantages</strong></td>
<td>Low computational complexity</td>
<td>Theoretically and computationally attractive</td>
<td>Find all possible p-mediands with the best answer</td>
</tr>
<tr>
<td><strong>Disadvantages</strong></td>
<td>Does not always give the best answer</td>
<td>Fail for some values of p</td>
<td>Time and memory consumption</td>
</tr>
</tbody>
</table>
Criteria for p-medians search in SoC

To find p-medians with the following criterias:

1) Distance
2) Throughput
3) Load of P-medians
4) Support for multiple criteria
Algorithm for p-medians search (1)

1. Enter input data
   - Topology of SoC
   - Rules of routing
   - Number of medians (p=2)
   - Distance requirement (d=1)

2. Create matrix of minimal distance
   ![Matrix of minimal distance]

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Algorithm for p-medians search (2)

3. Sort matrix of min distances according to distance requirement
   • All start nodes are potential medians
   • Nodes, which are marked in red, will not be considered (don’t satisfy the requirement)
   • Nodes that are in the same column under the start node are attachable nodes

4. Create median set Mm and add P potential medians to Mm
   For example, Mm={1,2}
Algorithm for p-medians search (3)

5. Create not median set $M_n$

6. For each node in $M_m$:
   - If a node from attachable nodes set doesn’t exist in $M_m$ or $M_n$ then this node is added to $M_n$
   - Sum distance from median node to not median node

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td>50</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>11</td>
<td>21</td>
<td>11</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>31</td>
<td>12</td>
<td>51</td>
<td>41</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>51</td>
<td>52</td>
<td>22</td>
<td>12</td>
<td></td>
</tr>
<tr>
<td>52</td>
<td>42</td>
<td>43</td>
<td>33</td>
<td>32</td>
<td></td>
</tr>
</tbody>
</table>

For example, $M_m=\{1,2\}$ $M_n=\{3,4,5\}$ sum =3

7. If $M_m$ and $M_n$ cover the total set of nodes in the graph $G$ then solution is added to solutions set

$$M_m \cup M_n = G$$

8. Result solution is the one with minimal sum value
Solutions

Input data:
P = 2
D = 1

Output data:
Sum = 3
Mm = \{1, 2\}  Mn = \{4, 3, 5\}
Mm = \{2, 5\}  Mn = \{1, 3, 4\}
Mm = \{3, 4\}  Mn = \{2, 1, 5\}
Throughput criteria in p-medians search

Algorithm is the same as the algorithm with limit on distance
Matrix of minimum throughput is used instead of distance matrix

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Link represented as [link load] / [link throughput]
Throughput matrix creation

Throughput is minimum value of links throughputs from shortest route

If there are several shortest routes, minimum value of throughput is found for each route. Maximum value is used.

Route 1: 1-3-2, \( \text{Thr}_{\min_1} = 3 \)
Route 2: 1-4-2, \( \text{Thr}_{\min_2} = 2 \)
Route 3: 1-5-2, \( \text{Thr}_{\min_3} = 1 \)

\[ T_{\text{thr}_{1,2}} = \max\{\text{Thr}_{\min_i}\} \]

\[ \text{Thr}_{1,2} = 3 \]
Load requirement

Load is analyzed after getting solutions with distance and/or throughput limit

Load of system = 800 Mbit/s
Load of 1 node = 300 Mbit/s
Load of 4 node = 500 Mbit/s
Average load of node = 400 Mbit/s

Delta load of 1 node = -100 Mbit/s
Delta load of 4 node = +100 Mbit/s

Developer
Multiple criteria in p-median search

Matrix min distance  Matrix min throughput

Applying distance and throughput requirements

Adapted matrix

Directed Tree Search

A set of solutions satisfying the distance and throughput requirements for nodes that are selected as P-Median

Analyze Load P-median

Result solution
Conclusion

Described method of directed tree search:

- For any value of $P$
- Always gives the best solution

Using $p$-medians for nodes allocation:

- Searching for components position on chip according to requirements (distance, throughput, load and etc.)
- It is possible to use several criterias together
Future work

1. Create methodology for searching for p-medians of other types
2. Analyze energy consumption
3. Add different routing algorithms