Parallel Placement for FPGAs Revisited

Cristinel Ababei
cristinel.ababei@ndsu.edu
ECE Department, North Dakota State University
Motivation

• Efficiency of classic sequential placement algorithms does not keep up with FPGAs size growth
• Simulated annealing based FPGA placement is the runtime bottleneck
• Traditional parallel distributed implementations suffer from communication and synchronization overheads
Solution: Parallelization Using Multithreading

Parallel Placement() {
  Step 1: Main thread
  4-way partitioning into subchips
  Subchips added to queue of tasks

  Step 2: Worker threads
  Place subchips until queue empty

  Step 3: Main thread
  Top-level low temp. SA refinement
}

Level 1 four-way partitioning

Level 2 part.
Solution Details

Step 1: 4-way partitioning

After Step 3
Results

Average: 20 testcases of VPR
Sequential VPR place vs. Parallel place using 4 threads

<table>
<thead>
<tr>
<th>Figure of merit</th>
<th>Parallel VPR place</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU place</td>
<td>2.5 X faster</td>
</tr>
<tr>
<td>WL place</td>
<td>+3.69 %</td>
</tr>
<tr>
<td>WL route</td>
<td>+2.19 %</td>
</tr>
<tr>
<td>Delay route</td>
<td>+2.15 %</td>
</tr>
</tbody>
</table>
Conclusion

• Parallel VPR placement based on multithreading:
  – 2.5 X speed-up using four threads
  – Minimal wirelength (WL) and delay degradation
  – Achieves better quality for a given runtime budget

Current and future work

• Testing on larger testcases of VPR 5.0. Seeking industry testcases
• Parallelizing routing step
• Download: http://www.ece.ndsu.nodak.edu/~cris