CS5160 FPGA Architecture & CAD
Final Project

111062625 蔡哲平

1. **How to compile and execute my program.**
   - Compile: Enter ***src/*** and make, it'll generate the executable file.
     ```
     $ cd src
     $ make
     ```
   - Execute
     ```
     $ ./topart [INPUT_FILE] [OUTPUT_FILE]
     ```
     e.g.
     ```
     $ ./topart ../benchmarks/B1.txt ../output/B1.txt
     ```

2. **The total topology constraints violations, the sum of external degrees, and the runtime of each testcase.**

	B1	B2	B3	B4	B5	B6	B7	B8
Topo. Viol.	19	78	368	597	4226	21668	42531	86261
Ext. Deg.	91	301	1432	2383	16964	86212	170757	345451
Runtime (s)	0.001	0.002	0.008	0.014	0.057	0.306	0.650	1.368

3. **The details of my implementation:**

   My program is referenced from the paper: "TopoPart: a Multi-level Topology-Driven Partitioning Framework for Multi-FPGA Systems", in Proc. of 2021 International Conference on Computer-Aided Design. However, coarsening and uncoarsening are not implemented in my program and I simplified other algorithms in this paper. I first perform a candidate FPGA propagation algorithm to find the candidate FPGAs for each node. Then, partitioning is conducted on the circuit graph based on the candidate FPGAs found in the previous step. Finally, for nodes that have no candidate FPGAs, assign it to an FPGA that hasn't reached maximum capacity.

   I. Candidate Propagation

   Initialize fixed nodes' candidate sets to contain its FPGA node only. As for other nodes, initialize their candidate sets to contain all FPGA nodes.

   II. Partition

   A priority queue Q is used to decide the order to assign the moveable nodes, in which the nodes with higher candidate FPGAs will be assigned with higher priority. For each moveable node extracted from Q, assign it to the first FPGA

node in its candidate set. If not assignable, remove the FPGA node from it and re-assign again. Finally, if any remaining node has zero candidates, assign it to a random FPGA that hasn't reached the maximum capacity.

4. What problems have you encountered with this homework?

At first, I aimed to implement the same algorithms in the paper. However, after some endeavors, I've found that it was quite complicated and will take a lot of time. Hence, I tried to simplify the algorithms and preserved only the essential parts. Originally, I thought that my result will be bad compared to other students since I abandoned lots of tricks applied in the paper. However, I consider my result to be acceptable because its runtime is extremely short and can come up with a decent result.