CS613200 Final Project

Due day: 06/25 2021

This project will implement a two-stage algorithm for technology mapping in FPGA design on a Linux environment.

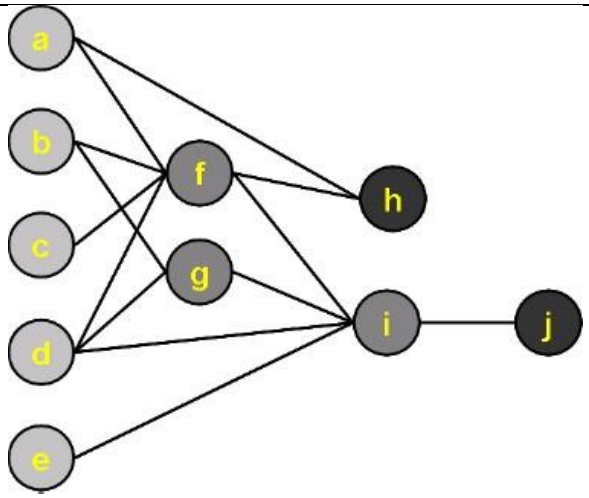
Stage1: Minimum Level Two-input Decomposition

A Boolean circuit is a network of Boolean gates, e.g., AND, OR, and NOT gates. The level of a circuit indicates the maximum number of gates from primary inputs to primary outputs. So, the circuit level further relates to the circuit delay.

A general circuit may consist of Boolean gates with more than two inputs. Replacing the multiple-input gates with smaller gates is widely adopted before the technology mapping, i.e., the decomposition. In the first stage, you will have a Boolean circuit that contains multiple-input gates. Your task is to decompose those gates into several two-input Boolean.

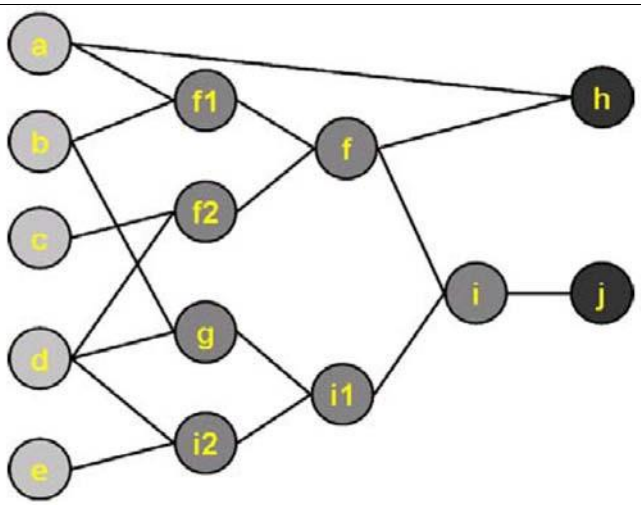
BLIF Example: test01.blif

```
.model sample01
.inputs a b c d e
.outputs h j
.names a b c d f
1--- 1
-1-- 1
--1- 1
---1 1
.names b d g
1- 1
-1 1
.names a f h
1- 1
-1 1
.names d e f g i
1111 1
.names i j
0 1
.end
```



Output BLIF Example: output.blif

```
.model sample01
.inputs a b c d e
.outputs h j
.names a b f1
1- 1
-1 1
.names c d f2
1- 1
-1 1
.names f1 f2 f
1- 1
-1 1
.names b d g
1- 1
-1 1
.names a f h
1- 1
-1 1
.names d e i2
11 1
.names g i2 i1
11 1
.names f i1 i
11 1
.names i j
0 1
.end
```



Stage2: Delay Optimal FPGA Technology Mapping

The FPGA comprises K-input lookup tables (LUTs) and Flip-Flops (FFs), which can simulate any circuit's functionality. The technology mapping is the step to map a circuit's functionality with the FPGA's LUTs and FFs. Since the LUTs are limited hardware resources on the FPGA, a good mapping uses fewer FPGA LUTs to achieve the functionality. (In this project, we only consider the K-input LUTs.)

With the decomposed circuit from the first stage, this stage's task is to map the circuit to a K-input LUT FPGA. You will write a program to perform LUT technology mapping on FPGA with a minimized circuit level and number of LUTs.

Input/output Format:

- Input file: A BLIF format file with 3 types of Boolean operations, i.e., AND, OR, and NOT.
- Standard output: The circuit level and number of LUT of your circuit.
- Output file: A BLIF format file of your circuit.

Input file example: map01.blif

```
.model map01
.inputs a b c d e f g h i j k l m n
.outputs o
.names [25] [26] o
11 1
.names k x1 x2
1- 1
-1 1
.names [19] [20] x3
1- 1
-1 1
.names n [18] x1
11 1
.names m l [18]
11 1
.names h g [19]
1- 1
-1 1
.names j i [20]
1- 1
-1 1
.names b a [21]
11 1
.names d c [22]
11 1
.names f e [23]
11 1
.names [22] [21] [24]
11 1
.names x3 [23] [25]
11 1
.names x2 [24] [26]
11 1
.end
```

Run-time Example:

```
%> ./map –k 4 map01.blif output.blif # map circuit by 4-input LUT
The circuit level is 2.
The number of LUTs is 5.
```

Output file example: output.blif

```
.model map01
.inputs a b c d e f g h i j k l m n
.outputs o
.names x2 x3 [23] [24] o
1111 1
.names k l m n x2
1--- 1
-111 1
.names g h i j x3
1--- 1
-1-- 1
--1- 1
---1 1
.names f e [23]
11 1
.names a b c d [24]
1111 1
.end
```

Notice:

1. The score considers the circuit level (primarily) and the number of LUT (secondary).
2. The output circuit's functionality must be equivalent to the input circuit. You may use the ABC command (i.e., cec) for equivalent checking!
3. LEDA is a C++ library for efficient data types and algorithms. You can download the static library file of LEDA from the course website and use the graph algorithm (e.g., min_cut) provided by LEDA API in your code. (http://www.algorithmic-solutions.info/leda_guide/graph_algorithms/mincut.html)
4. Please follow the format of run-time example:
   `%> ./map –k 4 map01.blif output.blif`
5. TA will test your program with different k-input sizes.
6. Please submit your source code, makefile, and a readme that describes how to compile your code.
7. A report is required (10%), which includes your data structure, algorithm, and results.
8. You must demo your code to TA orally. You get zero score for skipping demo.

Reference

[1] J. Cong, Y. Ding, An Optimal Technology Mapping Algorithm for Delay Optimization in Lookup-Table based FPGA Designs, IEEE Trans. on CAD, Vol. 13, No. 1, Jan. 1994, pp. 1-12

[2] Leda, http://www.algorithmic-solutions.info/leda_guide/Index.html

Any further question, please consult with TA: 孫勤昱

E-mail: suncy.2tanley@gmail.com Ext. 33573