

Programming Assignment 3-3:

DAGON Approach

DAGON approach:

step1: Partition subject graph into trees

step2: Optimally cover each tree (using dynamic programming.)

Object of this exercise:

Given a subject graph and a set of pattern graph in canonical representation (2-input NAND and INV), implement the second step of DAGON approach. (Both the subject graph and the pattern graphs are trees.)

Format:

Using **VLR representation** to represent the pattern graph and the subject graph to be covered. We use the following example to explain the VLR representation.

The VLR representation of the netlist in Fig. 2 is shown in Fig. 1.

NOTE: CIR_BEGIN, CIR_END, NAND2, INV and INPUT are reserved words.

NOTE: the VLR representation is case sensitive.

```
CIR_BEGIN
nand3.cir    ← netlist name
3            ← netlist cost
NAND2        ← represent the structure of netlist in VLR order.
INV
NAND2
INPUT
INPUT
INPUT
CIR_END
```

Fig. 1 The VLR representation of Fig. 2.

nand3.cir, cost=3

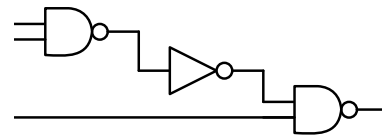


Fig. 2 A canonical representation of 3-input nand.

The VLR representation of a netlist is included in the keywords CIR_BEGIN and CIR_END. After the next two lines of the keyword CIR_BEGIN are the *netlist name* and the *netlist cost* respectively. After the netlist cost, the structure of the netlist is represented by traversing the tree in VLR order (**vertex first, left-child or top-child secondary, and right-child or bottom child last**), from netlist output to netlist input.

To make the matching easily to implement, the netlist in the Fig. 2 and the netlist in the Fig. 3 can be treat as difference in your matching. Putting each possible order of identical pattern graph in the set of pattern graph can solve this issue.

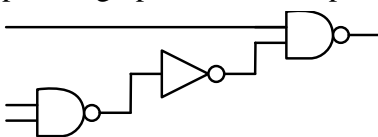


Fig. 3 Another order of 3-input nand.

Following netlists are used as pattern graphs and the file of its VLR representation is also provided that is named as “pattern_file”.

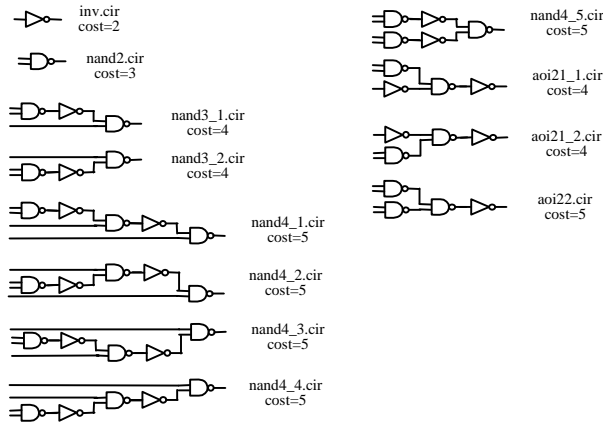


Fig. 4 the set of pattern graphs

Requirements:

The program must be able to receive commands in following format:

```
% a.out pattern_file subject_file
```

The file ‘pattern_file’ includes a set of VLR representation of pattern graphs and the file ‘subject_file’ includes the VLR representation of subject graph to be covered. You must **output the covered result in a file named ‘matched_netlist’**. In the file ‘matched_netlist’, **the cost and the structure of the netlist after matching** is also **represented in the format of VLR representation**. For example, Fig. 6 shows the matched output of Fig. 5 and recorded in the file “matched_netlist”. The VLR representation of Fig. 5 is put in the file “subject_file”.

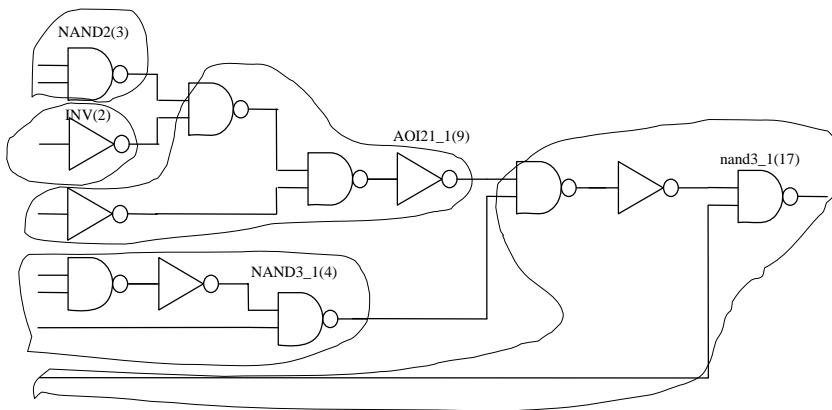


Fig 5. Optimal cover of a netlist named test.cir.

```

CIR_BEGIN
test.cir          ← netlist name
17                ← optimal cost
nand3_1.cir       ← structure of matched pattern graph
aoi21_1.cir
nand2.cir
INPUT
INPUT
inv.cir
INPUT
INPUT
nand3_1.cir
INPUT
INPUT
INPUT
INPUT
CIR_END

```

Fig. 6 The output format.

Grading:

Your grade depends on the output format, correctness, runtime and storage requirement. You may first compress all of the files, including the readme file, the source code, and the executable file, and then email your homework to TA. (please specify your student ID in the subject). A readme file that describes how you implement this exercise e.g., the flow chart ...etc., is also required.