Problem A: Learning Arithmetic Operations from Gate-Level Circuit

Chung-Han Chou, Chih-Jen (Jacky) Hsu, Chi-An (Rocky) Wu, and Kuan-Hua Tu
(Cadence Design Systems, Inc.)

Q&A

Q1. I have some questions about Problem A and need your help to check it.
The problem A said: Parallel computation with multiple threads or processes is not allowed.
Does it mean that we cannot use GPU? or cannot use machine learning?
Thank you very much.
A1. GUP is not allowed in this contest. However, you still can use machine learning algorithms by performing inference on a CPU.

Q2. I have a question about Problem A:
Do the designs only contain combinational logic? Thank you.
A2. Yes, there will only be combinational logic in the netlists.

Q3. We would like to know if there's any restriction on the problem size. Thanks.
(for example, the maximum number of gates or input/output nets)
A3. We do not make assumptions about the size of input netlist.

Q4. Will you provide a Verilog parser to Contestants to parse the netlists to be learned and make them concentrate on the learning algorithms?
A4. No, we won't provide Verilog parser to contestants.

Q5. We found a way to rewrite any given circuit with constant cost (5). We would like to know if this is a valid solution and if the cost is really 5. Thank you for your consideration.

Given a combinational circuit,
we can represent the circuit using only NAND gates.
To rewrite the circuit with verilog operators,
we can simply concatenate the fanin nets of every NAND gate into 2 bit vectors,
and use the bit-wise NAND operation to complete the whole circuit.

The verilog code will look like the following,

```
assign { z1, z2, ..., zn } = ~( { x1, x2, ..., xn } & { y1, y2, ..., yn } );
```

and the total cost would be 5.

A5. Thanks for pointing out the shortcomings. We modify the cost of logic operators and concatenations to reflect bit-width or symbol number. Please refer to the updated problem description released on the website.

Q6. We have a question about the format of input file of problem A.
Could we assume the format of the input verilog files would be as same as the sample input files as follows:

module top(<inputs>, <outputs>);
    primary input declaration;
    primary output declaration;
    wires for primary inputs (same order with input declaration);
    wires for primary outputs (same order with output declaration);
    intermediate wires declaration;
    gate-level circuits;
endmodule;

and will primary inputs named after in1, in2, in3.....
primary outputs named after out1, out2, out3.....?

It would be a big help on parsing the input files if we can make the assumption.
Thank you very much!

A6. Contestants are not allowed to make any assumption on variable names of input/output ports, internal wires, and gate names.

Q7. Should we consider 2-value logic or 4-value logic in this problem? Or can it be promised that they make no difference? Thank you.

A7. Contestants only need to consider 2-value logic. Each signal can only be true or false.

Q8. For problem A, I would like to ask that whether there would be hidden testcases in Beta Test and Final submission? Thank you.

A8. There will be hidden cases in final evaluation.

Q9. Could you please help to advise if we can use other auxiliary software for the problem? Thanks for giving help.

A9. Participators can find auxiliary software such as abc by themselves.

Q10. Currently we use python3 to write the main program, and we use “yosys” and “abc” to support the main program. We have tried to use “pyinstaller” to transfor .py file into executable binary file. But when we ran the binary file on the TSRI machine, an error occurred, “Error loading Python lib ‘/tmp/_MEIV8idBc/libpython3.8.so.1.0’: dlopen: /lib64/libm.so.6: version ‘GLIBC_2.29’ not found (required by /tmp/_MEIV8idBc /libpython3.8.so.1.0)”. When we enter the command “python3 cada0027.py”, then we can run the .py file successfully.
So, could you help to check if we have to submit the binary file? or Can we submit the .py file?
A10. The .py file can be accepted.
The argument should still follow the given format ‘<executable.py> -input <in_file> -output <out_file>’.

Q11. 我們想請問：最終生成的 verilog 檔是否要求"可合成"的，還是只需要"可模擬"的就可以？謝謝。
A11. The output Verilog RTL should be synthesizable.

Q12. For problem A, I would like to know that for the final submission, has the percentage/amount of public testcases and hidden testcases determined? Thank you.
A12. We may have 50% hidden cases in the final evaluation.

Q13. If we find an equation like this, out1 = in1*in1*in1, then can we transform it as out1 = in1**3? The result for the beta test is ok, but it is NOT synthesizable while we try to run it.
A13. We accept only const ** const and var ** const in this contest, which are synthesizable. As the result, in1 ** 3 is allowed.
However, syntax like var ** var is not synthesizable. For example, in1 ** in2 is not allowed.

Q14. 這題的 out1 我們輸出的是 in1 ** 3，在 beta test report 結果是成功 pass 此問題，但在跑 spyglass 這個 tool 時顯示無法合成。想請問 ** 這個運算式是被允許的嗎？
A14. We accept only const ** const and var ** const in this contest, which are synthesizable. As the result, in1 ** 3 is allowed.
However, syntax like var ** var is not synthesizable. For example, in1 ** in2 is not allowed.

Q15. For problem A, I know that the keyword "*signed*" is allowed. I would like to know if the keyword "*$signed()*" is also allowed. Thank you.
A15. Yes, $signed() is allowed.