Problem Set #6

The **due date** for this homework is **Tue 23 Apr 2013 12:59 PM EDT -0400**.

Question 1

Tech Mapping: Basics

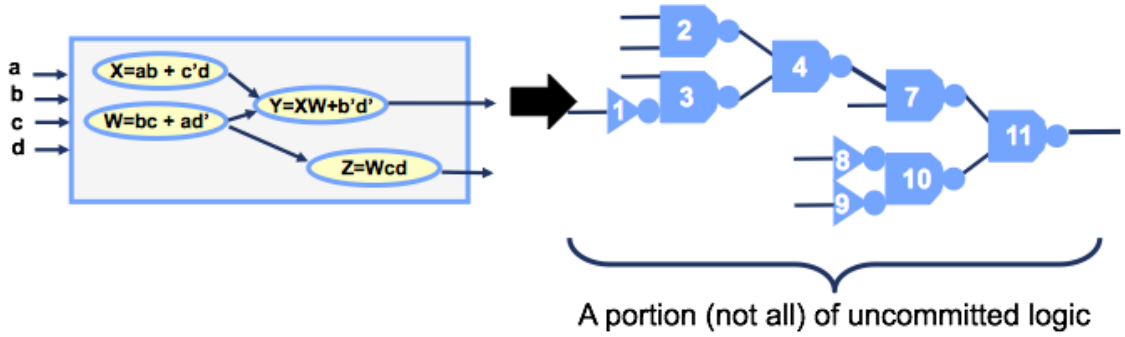
Which of these are correct statements about technology mapping?

- [ ] The tree to be covered is called pattern tree, and a gate in the technology library is called a subject tree.
- [ ] The input to technology mapping is called uncommitted logic, because the netlist is represented as simple gates – NAND and NOT – which are not the final gates in our technology library.
- [ ] The Boolean network obtained from synthesis step is usually transformed to a flat network of NANDs and NOTs before the technology mapping.
- [ ] Although the tree-covering algorithm cannot be applied to a netlist in DAG (directed acyclic graph) form, it can find the global optimal covering after treeifying the graph.
- [ ] After technology mapping, the netlist is expressed in a set of pre-designed gates from a technology library, and it is technology dependent.

Question 2

Uncommitted logic

Because the output of the multi-level synthesis process is not “real” gates from your technology library, the first step in the tech mapping process is to transform the Boolean network model into uncommitted logic, made of simple real gates. Consider the small Boolean logic network show below, and a portion (not all) of the uncommitted logic we propose to use to represent this network. Which of these are correct statements about this network and this logic?

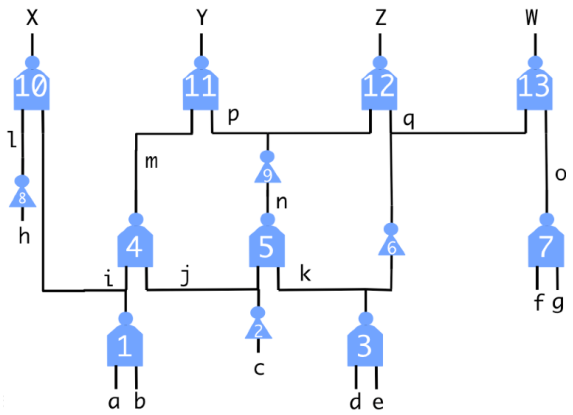


- [ ] This uncommitted logic would correctly implement the W and Y nodes of the Boolean logic network.
- [ ] This uncommitted logic would correctly implement the W and Z nodes of the Boolean logic network.
- [ ] This uncommitted logic would correctly implement the Y and Z nodes of the Boolean logic network.
- [ ] This uncommitted logic would correctly implement the X and W nodes of the Boolean logic network.
- [ ] This uncommitted logic would correctly implement the X and Z nodes of the Boolean logic network.

Question 3

Tree-ification of the Input Uncommitted Logic

Consider this network and NAND and NOT gates, which is the uncommitted logic we need to tech map. As we explained in the lecture, our mapping algorithm requires us to start with trees, not DAGs. Which of the following are correct statements about this uncommitted logic network:



- [ ] We must cut the wire labeled q to tree-ify this network.
- [ ] It is not possible to tree-ify this network.
- [ ] This network will tree-ify in 3 separate trees, each of which we can map with our tech mapping algorithm.
- [ ] We must cut the wire labeled i to tree-ify this network.
- [ ] We must cut the wire labeled a to tree-ify this network.

Question 4

Library Patterns that Cannot Be Tree-fied

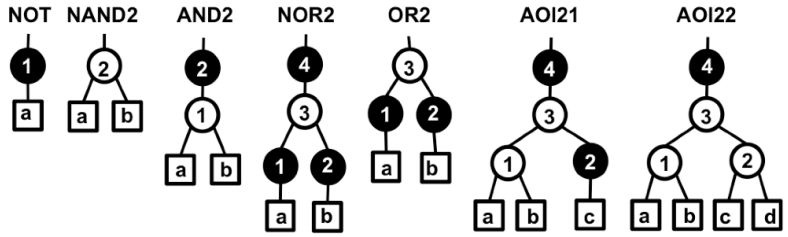
We noted in the lecture that some “simple” logic elements that we might expect to appear in a typical standard cell library are not allowed, just because their representation in the simple NAND2-NOT style cannot be tree-ified. (Yes – there are tricks to do these, but for this question, we will focus only the tree-based algorithms we taught in the lecture.) We mentioned one such logic gate in the lecture. Are there any others? Consider these simple logic elements that might appear in a technology library. Which of these are correct statements about these logic elements?

- [ ] A 3-input EXNOR gate (output Z = (a⊕b⊕c)') can be tree-ified successfully.
- [ ] A 2-input Multiplexor (MUX) element (output Q = D0•select’ + D1•select) cannot be treeified.
- [ ] A 3-input NAND gate (output Z = (abc)') cannot be tree-ified.
- [ ] A 2-input EXNOR gate (output Z = ab + a’b’) cannot be tree-ified.
- [ ] A 3-input NOR gate (output Z = (a+b+c)') cannot be tree-ified.

Question 5

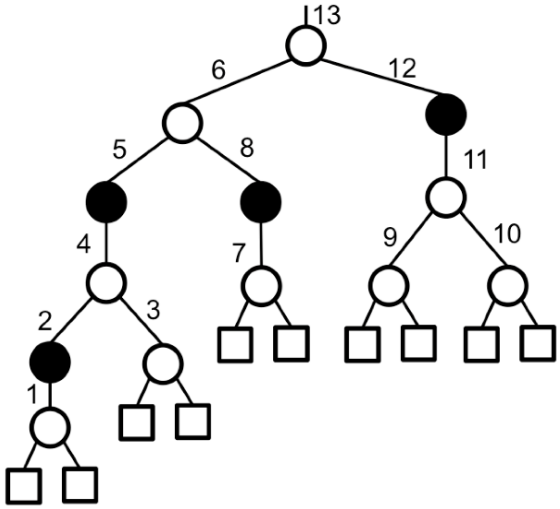
Library Matching in the Subject Tree

Consider again the technology library we introduce in the lecture, with 7 gates, from simple (NOT, NAND, AND, OR, NOR) to complex (AOI)



(Note: the numbers labeled on the nodes are for visually impaired students.)

Here is a new subject tree, built from NAND2 and NOT gates.



We label the internal wires with numbers. In this problem, we want you to execute (by hand) the **matching procedure**, and annotate the network so that you know, for every node, what library pattern matches. For clarity we will refer to the node by the number on the wire **above** it, i.e. the name of output of this node. So, for example, we can match a NOT pattern at node 8 in this network, and we can also map the AND2 pattern at node 8.

Which of these are correct statements about matching from the technology library to this subject tree?

- [ ] The AND2 pattern from the library matches the subject tree at node 12.
- [ ] The OR2 pattern from the library matches the subject tree at node 6.
- [ ] The AND2 pattern from the library matches the subject tree at node 8.
- [ ] The NOR2 pattern from the library matches the subject tree at node 5.
- [ ] The OR2 pattern from the library matches the subject tree at node 3.

Question 6

Recursive mincost covering: formulation

Consider again the technology library with 7 gates, and the subject tree with 13 internal labeled wires, from Q5. We showed in lecture that there was a surprisingly simple recursive formulation for the minimum cost covering problem, that was enabled by our restriction of all target and subject netlists to be trees. We showed in detailed example in lecture how to write the recursive equation for **mincost(node)**, for any node of the subject tree, if we assign costs to each pattern in our technology library. Suppose we assign these costs to the library patterns:

- NOT: cost = 1
- NAND2, NOR2: cost = 2
- AND2, OR2: cost = 4
- AOI21, AOI22: cost = 4

Which of these will appear during the execution of the recursive mincost matching algorithm, as it visits each node of the tree and recursively computes the minimum cost matching?

- ☐ mincost(13) = 2 + mincost(6) + mincost(12)
- ☐ mincost(8) = 1 + mincost(7)
- ☐ mincost(4) = 2 + mincost(1) + mincost(2) + mincost(3)
- ☐ mincost(5) = 4 + mincost(1) + mincost(3)
- ☐ mincost(12) = 4

Question 7

Final Mincost Cover

Consider again the technology library with 7 gates and gate costs form

Question6, and the subject tree with 13 internal labeled wires, from Q5. In Q5, you determined what library patterns match the subject tree. In Q6, you explored how the **mincost(node)** recursion works in the tree. To finish, in this problem, we want you to fully execute (by hand) the mincost covering recursion. Do this as in the lecture: make a small table that has a row for each of the 13 labels in the subject tree. For each library pattern that matches that at that subject node, write the **mincost(node)** recursive equation. Fill out the table as you work through the recursion. Compute the final, best mapping of the subject tree.

Which of these are correct statements about the final, best cost mapping of this subject tree?

- [ ] In the final cover (table), we choose a OR2 gate at subject tree node 6 with mincost(6)= 15.
- [ ] The best cost cover of the tree is 20, and it has 9 total library gates.
- [ ] The best cost cover of the tree is 15, and it has 6 total library gates.
- [ ] In the final cover (table), we choose a NAND2 gate at subject tree node 10 with mincost(10)= 2.
- [ ] The best cost cover of the tree is 16, and it has 7 total library gates.
- [ ] In the final cover (table), we choose a NAND2 gate at subject tree node 3 with mincost(3)= 2.
- [ ] In the final cover (table), we choose a NOT gate at subject tree node 12 with mincost(12)= 7.
- [ ] In the final cover (table), we choose a NAND2 gate at subject tree node 13 with mincost(13)= 16.
- [ ] In the final cover (table), we choose a AND2 gate at subject tree node 2 with mincost(2)= 4.
- [ ] In the final cover (table), we choose a NOT gate at subject tree node 2 with mincost(2)= 3.

- [ ] In accordance with the Honor Code, I certify that my answers here are my own work.

Submit Answers Save Answers