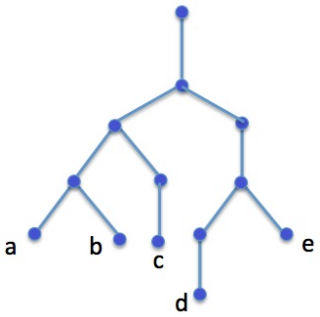
CSEE E6861y **Homework 4** Handout 28
Prof. Steven Nowick March 29, 2016

This homework is due at the *beginning of class* on Thursday, April 7.
NOTE: A correct answer without adequate explanation or derivation will have points deducted. To get full credit, (a) write legibly/type, and (b) show all work (label relevant items, show derivations, include explanations).

1. (20 points) **Technology Mapping: Representation Using Base Functions.**
   (a) **Subject Graphs.** You are given the Boolean function $f = (a + bc') \cdot (d + e + cf')$. *Write two distinct subject graphs* for this function, i.e. which are not isomorphic, using only the base functions NAND2 and INV. As usual, in basic technology mapping, the subject graphs should have no double inversions. *Hint:* first, derive two distinct gate-level decompositions of this function into equivalent circuits using only NAND2 and INV gates, then draw the two corresponding subject graphs.
   (b) **Pattern Graphs.** Write the pattern graphs for two cells, *(i) NAND3* and *(ii) OAI22*, using only NAND2 and INV. An OAI22 (OR-AND-INVERT-22) gate is a compact VLSI cell (somewhat analogous to the AOI22 cell in De Micheli fig. 10.8), which combines two OR2's feeding into a 2-input NAND. The gates are drawn merged together, since the VLSI implementation is a single complex CMOS gate (compare to the AOI22 cell in fig. 10.8 in De Micheli).
   Note: draw *all* possible distinct (i.e. non-isomorphic) pattern graphs for each of (i) and (ii). That is, if there are two or more pattern graphs for each cell with different structure, draw each such graph. (For example, see De Micheli Fig. 10.7(c) and (d) for two non-isomorphic pattern graphs for AND4.)

2. (25 points) **Basic Area-Oriented Technology Mapping: Tree Covering Algorithm.**
   For this problem, you are to do technology mapping for a simple combinational design, using the basic procedure presented in class and handouts, and the reading for minimizing area. For function f, use the subject graph given below. For the cell library, use the pattern graphs given in De Micheli, fig. 10.8. Assume the costs (i.e. areas) of the library cells are as follows: INV=2, NAND2=3, NOR2=3, AND2=4, OR2=4, AOI21=4, AOI22=5.
   You will first do basic area-oriented technology mapping, as covered in Handout #24 and in class lecture and reading. You will then apply an advanced extension, the *inverter-pair heuristic,* aiming at improved cost.



   (a) (10 points) **Basic Area-Oriented Tree-Covering.** Follow the basic dynamic programming tree-covering algorithm in Handout #24 (and in De Micheli, pp. 522-523): (i) *construct a table,* with identical columns as in Handout #24, which indicates the cell matches and total cost, for each node in the subject tree, in a bottom-up traversal; (ii) *indicate* the cost of the final optimal cover; and (iii) *draw* the final optimal tech-mapped implementation. (*Note:* Determine the matches at each node by inspection; do not use the MATCH algorithm.)
   (b) (15 points) **Advanced Technique: Inverter-Pair Heuristic.** For this problem, you will be using the subject graph shown above as a starting point. You will then use the *inverter-pair heuristic* to enhance the subject graph, to enable a better solution.
   First, carefully read in De Micheli, ch. 10.3.3, p. 530 bottom to p. 532 top (Figure 10.17) on the basic inverter-pair heuristic method. Then do the following:

(i) *Enhanced Subject Graph.* Draw the enhanced subject graph for the inverter-pair heuristic, following the method in the above reading. In particular, extra inversions should be added at internal nodes and leaves, precisely as indicated in the De Micheli reading and in Figure 10.17. However, there should be *no extra inversion* added at the root. In addition, do *not expand* internal single inversions into triple inversions, keep as single inversions. Otherwise, follow all guidelines presented in this reading.

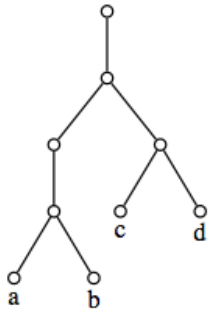
(ii) *Enhanced Pattern Graphs.* Show any new pattern graphs that need to be added to the given cell library to support the inverter-pair heuristic, following the method in the above reading. The resulting library will include all original pattern graphs, plus some addition ones.

(iii) *Tree-Based Covering.* Given the above, **by inspection,** draw the minimum-cost final covering of the enhanced subject graph by pattern graphs from the augmented cell library (based on steps (i) and (ii) above). Only show the final min-cost covering of the subject tree. Also, *indicate the final area cost* of your solution.

3. (30 points) **Advanced Delay-Oriented Technology Mapping: Load-Binning.** In this problem, you are to perform technology mapping on a given subject graph, oriented towards delay. You will use a more sophisticated delay model, considering the output load seen by each gate as a parameter for cell delay, as well as an enlarged library including cells with different drive strengths. You will also use the load-binning technique presented in class and in Handout #26.

Setup.

Subject Graph. The figure below shows the subject graph for a small circuit to be mapped.



Cell Library. We will start with the basic cell library presented in De Micheli, fig. 10.8. However, we will add some more cells to the library, and use a more sophisticated cost model for cell delay. The library is augmented as follows:

(a) **INV: 3 cells.** Assume *three* different INV cells: **INVw, INVs, INVds**. INVw has weak drive strength, but provides less load to its input. INVs has relatively strong drive strength, but provides more load to its input. INVds (i.e. double-strength) has strongest drive strength, but provides even more load to its input. Each cell has "intrinsic delay" $\delta_{INV} = 2$. (See below for the complete formula to compute the cell's overall delay.)

(b) **NAND2: 2 cells.** Assume *two* different NAND2 cells: **NAND2w, NAND2s**. (Note: here, and in remaining cells, 'w' means weak drive strength, and 's' means strong drive strength.) Each cell has intrinsic delay $\delta_{NAND2} = 3$.

(c) **AND2: 2 cells.** Assume *two* different AND2 cells: **AND2w, AND2s**. Each cell has intrinsic delay $\delta_{AND2} = 4$.

(d) **NOR2: 2 cells.** Assume *two* different NOR2 cells: **NOR2w, NOR2s**. Each cell has intrinsic delay $\delta_{NOR2} = 3$.

(e) **OR2: 2 cells.** Assume *two* different OR2 cells: **OR2w, OR2s**. Each cell has intrinsic delay $\delta_{OR2} = 4$.

(f) **AOI21: 1 cell.** Assume *only one* AOI21 cell. This cell has intrinsic delay $\delta_{AOI21} = 4$.

(g) **AOI22: 1 cell.** Assume *only one* AOI22 cell. This cell has intrinsic delay $\delta_{AOI22} = 5$.

Computing Cell Delay. Assume each cell c has a **cell delay** D_c given by the formula:

$$D_c = \delta_c + k \cdot L,$$

where δ_c is the basic *intrinsic delay* of the cell (listed above), k is its *drive parameter*, and L is the actual *output load* of the gate that the cell's output is driving. For the above library, assume that the **drive parameter** k for each **w cell is 1**, and the **drive parameter** k for each **s cell is 0.5**. For the special high-drive inverter, assume that the **drive parameter** k for this **ds cell is 0.25**. Finally, assume that the **drive parameter** k of **AOI21 and AOI22 cells is 1.**

Also, assume that **each w cell** *provides a load of 1* to its input gate, and **each s cell** *provides a load of 2* to its input gate. For the special high-drive inverter, assume **this ds cell** *provides a load of 4* to its input gate. Finally, assume that **the AOI21 and AOI22 cells** each *provide a load of 1* to their input gates.

Finally, assume a **load on the circuit's primary output of 8.**

Load Binning. Given the above assumptions, each gate can provide either a load of 1, 2 or 4 to the gate that drives it. Hence, the "load binning" you will use is to assume three bins: a load of 1, a load of 2, and a load of 4. *Note:* When you complete the algorithm and get to the primary output, the load on the final output gate is special: it is given as 8, and you should use this output load for computing the delay of the final output gate.

What to Do:

(a) Using the above delay formula, along with the given intrinsic delays, loads, and drive parameters for each cell, follow the tree-based covering algorithm, using dynamic programming, to cover the given subject graph. As in class, use the cell library of De Micheli, Fig. 10.8, with costs enumerated above. Follow the tree-covering approach for load binning as presented in class, and in Handout #26.

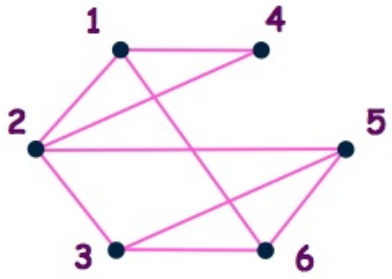
Derive the dynamic programming table, and clearly show all work.

(b) **Final Solution.** *Draw* the final optimal mapped circuit, and *indicate* its worst-case delay (i.e. its cost).

4. (25 points) **Optimal Circuit Bi-Partitioning.**

In this problem, you are to apply the Kernighan-Lin algorithm to optimally bi-partition a small circuit example.

You are given the small graph below with 6 nodes, representing interconnected gates in a circuit.



What To Do: Apply the Kernighan-Lin (KL) algorithm to this example to produce an optimal bi-partition of the nodes. **Assume the following initial "seed" bi-partition of the nodes into sets A and B:** $A = \{1,2,3\}$, $B = \{4,5,6\}$. Using this seed, follow the method presented in the assigned reading of the Dunlop/Kernighan paper and in the KL example handout. The result of the procedure should be two sets, A' and B', with half the nodes in A' and half the nodes in B', which is the result of following the complete KL method.

For full credit, show all work. Clearly indicate all E_i, I_i, D_i values and gain computations of each step, which swap is selected per step, and how a final result is derived.