CSEE E6861y Handout #26
Prof. Steven Nowick March 24, 2016

Technology Mapping: Delay Optimization and Load Binning

In this handout, the problem of delay optimization with "load binning" in tech-map is covered. This material was partly presented in class and in De Micheli, pp. 525-526. The model targets the handling of cell drive strengths and output loads for more accurate and realistic results. Dynamic programming is used, as with the area-oriented problem, to optimally solve the mapping problem.

Delay Model

A library cell *C* has delay:

$$D_c = \delta_c + kL \qquad \text{(see De Micheli, pp. 525-526)}$$

where δ_c is the ***intrinsic delay*** of the cell; L is the capacitive ***output load*** that the cell drives; and k is a constant, called its ***drive parameter***. This formula defines how a cell's actual delay depends on the load that it has to drive and its inherent drive strength. When k is small, the cell has good drive strength, i.e. its delay increases slowly with increasing output load. When k is large, the gate has poor drive strength. In each case, the delay also increases linearly with the output load that it must drive.

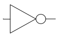
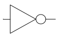
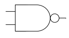
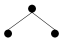
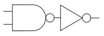
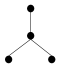
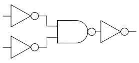
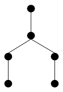
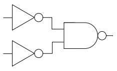
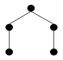
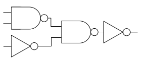
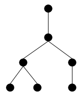
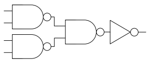
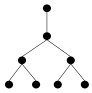
In addition, one is typically given the **load on the primary output gate**, i.e. the external load that the circuit's output drives.

Finally, a cell provides an ***input load***, which is the fixed capacitive load that it provides in turn to the input cells that drive it. Often, gates with larger drive strength (i.e. smaller k) have larger loads that they present to their input gates.

Cell Library

For this handout, we assume a simple extension of the cell library in fig. 10.8 of De Micheli, p. 516. For INV, we now include 2 distinct inverter cells, each with a different drive strength and input load. For all other gates, there will be one library cell each (as in De Micheli), but we indicate the new delay parameters. In general, of course, a realistic library has many cells for each gate type.

The library:

Match Name	Cell Name and Decomposition		Pattern Graph	Drive Parameter *k*	Input Load	Intrinsic Delay
t1A	INV A			1	1	2
t1B	INV B			0.5	2	2
t2	NAND2			1	1	3
t3	AND2			1	1	4
t4	NOR2			1	1	3
t5	OR2			1	1	4
t6	AOI21			1	1	4
t7	AOI22			1	1	5

Load binning: since there are 2 possible integer values for the input load of a gate, we use 2 load bins (1 and 2) as possible output loads during tree covering.

Primary output load: As a special case, at the primary output, **we assume a load of 8**. This load will not be in a pre-defined bin, and will be handled separately.

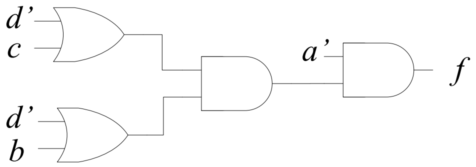
Goal: perform optimal delay-oriented tech-map, i.e. to obtain a minimized worst-case, i.e. "critical", input-to-output path.

Input arrival times: assume all inputs arrive at the same time: $t = 0$.

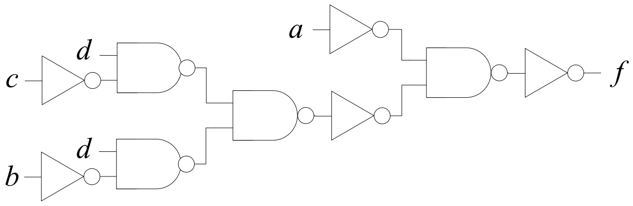
Example

Function: $f = a'((d' + c)\cdot(d' + b))$

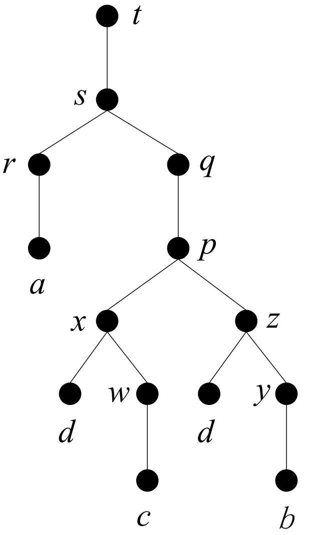
Initial circuit:



Initial decomposed circuit (using NAND2 and INV):



Corresponding subject graph:



Optimal Tree Covering Using Load Binning

index	vertex	match	gate	cost (gate delay)	assumed output load	min. total cost (i.e. delay) so far
1a	*w*	*t1A*	INV A	2 + 1 * 1 = 3 ①	[1] ②	3
1b				2 + 1 * 2 = 4	[2] ②	4
1c		*t1B*	INV B	2 + 0.5 * 1 = 2.5	[1]	$\boxed{2.5}$ ③
1d				2 + 0.5 * 2 = 3	[2]	$\boxed{3}$ ③
2a	*y*	*t1A*	INV A	2 + 1 * 1 = 3	[1]	3
2b				2 + 1 * 2 = 4	[2]	4
2c		*t1B*	INV B	2 + 0.5 * 1 = 2.5	[1]	$\boxed{2.5}$
2d				2 + 0.5 * 2 = 3	[2]	$\boxed{3}$
3a	*x*	*t2*	NAND2	3 + 1 * 1 = 4	[1]	4 + cost(*1c*) = $\boxed{6.5}$ ④
3b				3 + 1 * 2 = 5	[2]	5 + cost(*1c*) = $\boxed{7.5}$
4a	*z*	*t2*	NAND2	3 + 1 * 1 = 4	[1]	4 + cost(*2c*) = $\boxed{6.5}$
4b				3 + 1 * 2 = 5	[2]	5 + cost(*2c*) = $\boxed{7.5}$
5a	*p*	*t2*	NAND2	3 + 1 * 1 = 4	[1]	4 + max(cost(*3a*),cost(*4a*)) = $\boxed{10.5}$
5b				3 + 1 * 2 = 5	[2]	5 + max(cost(*3a*),cost(*4a*)) = $\boxed{11.5}$

index	vertex	match	gate	cost (gate delay)	assumed output load	min. total cost (delay) so far
6a	*q*	*t1A*	INV A	2 + 1 * 1 = 3	[1]	3 + cost(*5a*) = 3 + 10.5 = 13.5 ⑤
6b				2 + 1 * 2 = 4	[2]	4 + cost(*5a*) = 4 + 10.5 = 14.5
6c		*t1B*	INV B	2 + 0.5 * 1 = 2.5	[1]	2.5 + cost(*5b*) = 2.5 + 11.5 = 14 ⑥
6d				2 + 0.5 * 2 = 3	[2]	3 + cost(*5b*) = 3 + 11.5 = 14.5
6e		*t3*	AND2	4 + 1 * 1 = 5	[1]	5 + $\underline{\underline{\text{max}}}$(cost(*3a*),cost(*4a*)) = 11.5
6f				4 + 1 * 2 = 6	[2]	6 + $\underline{\underline{\text{max}}}$(cost(*3a*),cost(*4a*)) = 12.5
6g		*t7*	AOI22	5 + 1 * 1 = 6	[1]	6 + $\underline{\underline{\text{max}}}$(cost(*1c*),cost(2c)) = $\boxed{8.5}$
6h				5 + 1 * 2 = 7	[2]	7 + $\underline{\underline{\text{max}}}$(cost(*1c*),cost(2c)) = $\boxed{9.5}$
7a	*r*	*t1A*	INV A	2 + 1 * 1 = 3	[1]	3
7b				2 + 1 * 2 = 4	[2]	4
7c		*t1B*	INV B	2 + 0.5 * 1 = 2.5	[1]	$\boxed{2.5}$
7d				2 + 0.5 * 2 = 3	[2]	$\boxed{3}$
8a	*s*	*t2*	NAND2	3 + 1 * 1 = 4	[1]	4 + $\underline{\underline{\text{max}}}$(cost(*7c*),cost(*6g*)) = $\boxed{12.5}$
8b				3 + 1 * 2 = 5	[2]	5 + $\underline{\underline{\text{max}}}$(cost(*7c*),cost(*6g*)) = $\boxed{13.5}$
8c		*t5*	OR2	4 + 1 * 1 = 5	[1]	5 + cost(*5a*) = 15.5
8d				4 + 1 * 2 = 6	[2]	6 + cost(*5a*) = 16.5
9a	*t* ⑦	*t1A*	INV A	2 + 1 * $\boxed{8}$ = 10	[8]	10 + cost(*8a*) = 22.5 ⑧
9b		*t1B*	INV B	2 + 0.5 * $\boxed{8}$ = 6	[8]	6 + cost(*8b*) = $\boxed{19.5}$ ⑨
9c		*t3*	AND2	4 + 1 * $\boxed{8}$ = 12	[8]	12 + $\underline{\underline{\text{max}}}$(cost(*7c*),cost(*6g*)) = 12 + 8.5 = 20.5
9d		*t4*	NOR2	3 + 1 * $\boxed{8}$ = 11	[8]	11 + cost(*5a*) = 11 + 10.5 = 21.5

NOTES:

① On the left side of the cost equation, the first term stands for the intrinsic delay of the corresponding gate of the row, which is INV A in this case. The second term is a multiplication of two numbers: the first number stands for the drive parameter of the same gate; the second number stands for the assumed output load, which is also shown in the next column.

② Since there are two load bins for this problem, [1] and [2], there is a separate calculation assuming each in a distinct row for the same cell.

③ The circled numbers in this column are the best total cost so far at the current vertex, one best result for each load bin.

④ NAND2 provides an input load of 1, hence *3a/3b* ONLY use the best "*w*" match having output load bin of 1, which is *#1c* (i.e. *#1d is* not used here!).

⑤ When matching INV A at vertex *q* (i.e. index *6a/6b*), it has an input load of 1, hence it supplies an output load of 1 to the predecessor gate at *p*, so use *#5a* for vertex *p*.

⑥ However, when matching INV B at *q* (i.e. index *6c/6d*), it supplies an output load of 2, so use *#5b* for vertex *p*.

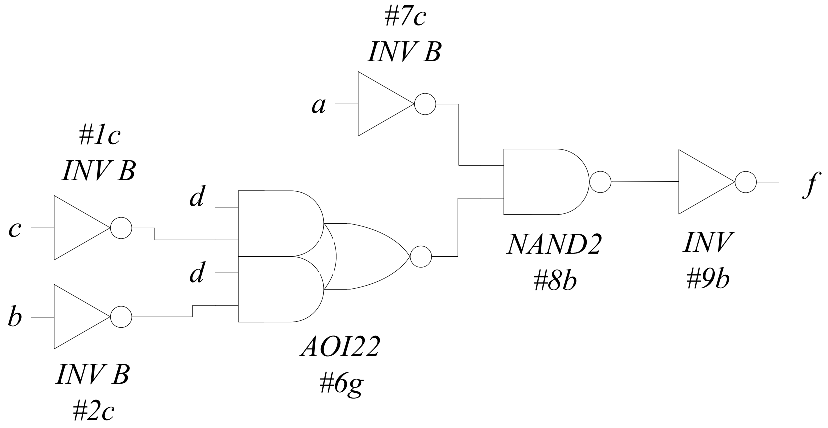
⑦ Vertex *t* is unique, since it is the root of the entire subject graph, hence it is the primary output of the circuit to be mapped. Use the given fixed output load of 8 (also see the column of "assumed output load'); no load binning is necessary.

⑧ The cell of the current row, INV A, supplies an output load of 1 to its input gate, so use #8a.

⑨ The gate of the current row, INV B, supplies an output load of 2 to its input gate, so use #8b. Also, the "best result so far" for root vertex t is also the ***final best delay-oriented mapping*** of the *entire* subject graph.

Final delay-optimal mapped circuit:

Recover the complete delay-optimal mapping from the above table, starting from the best solution at *#9b*, tracing backwards through corresponding previous best mappings that it used. The result is shown below:



Total Worst-Case (i.e. Critical Path) Delay = 19.5