A clarification about total path delay:

I would like to clarify the confusing point from lecture today, specifically my mistake that resulted in the error that several of you noticed: \(NF^{1/N} \neq F\)

The point of clarification is that the equation \(D = F + P\) is \textbf{wrong}. However, the other D equation: \(D = NF^{1/N} + P\) is correct. Let me explain the inconsistency:

Many of the gate level parameters (F, G, H, B) have the same relationship at the path level that they do at the gate level. For example, in the path effort \(F\):

\[F = GHB = \prod g_i h_i b_i\]

You can just substitute the small letters for big ones, and the relationship at the stage level for effort: \(f_i = g_i h_i b_i\) is transformed into the path effort \(F = GHB\)

Delay, on the other hand, is the sum of all of the individual delays, not the product of them. The delay of a single stage (gate) in a path is given by:

\[d_i = f_i + p_i = g_i h_i b_i + p_i\]

However, the path D is \textbf{NOT} \(D = F + P\), but rather:

\[D = \sum d_i = \sum (f_i + p_i) = Nf^\hat{\cdot} + P = NF^{1/N} + P\]

The confusion in class was a result of my incorrect assumption that the individual stage and total path delays were related in the same way to the individual stage and total path efforts. This is due to the \(D = \sum d_i\) condition, which requires that the efforts of individual stages be summed:

\[\sum f_i = \sum g_i h_i = g_1 h_1 + g_2 h_2 + \cdots + g_N h_N\]

rather than the product relationship that exists for the total path effort:

\[F = \prod f_i = \prod g_i h_i = (g_1 h_1)(g_2 h_2) \cdots (g_N h_N)\]
When the stages are optimized with the optimized stage effort $\hat{f} = F^{1/N}$ they still multiply together to give the product $F$. The value of $F$ does not change by changing and optimizing the stage efforts. Consider that the individual logical efforts $g_i$ (based on type of gate) and the path electrical effort $H = \frac{C_{last}}{C_{first}}$ are fixed, and therefore $F=GH$ is also fixed. Rather, the optimized stage effort reduces the total delay through the path by equalizing the effort produced at each stage in the path, thus minimizing the sum $\sum f_i$, which can change, and is at a minimum when all the stages have the same $f$. This is one of the goals of Logical Effort techniques, to optimize the delay by equalizing the efforts through a given path.

There is one other correction I would like to make, on branching effort. The branching effort $b_i$ is the number of gate inputs an output drives. I mistakenly said that it depended on the number of inputs a gate has (like a NAND3 gate would have $b = 3$ and an inverter would have $b = 1$. That is wrong.) however, the output is only connected to a single input of the next gate, so it doesn’t make a large difference how many inputs the gate has, at least for the purpose of simple comparisons. As such, the example where I crossed out a gate:

![Diagram](image)

was correct without the gate being crossed out. The first stage has $b_1 = 3$, because it is connected to 3 inputs, rather than one, and so it takes 3 times the effort to drive the next stage. (It is merely a coincidence that the gates also have 3 inputs.)