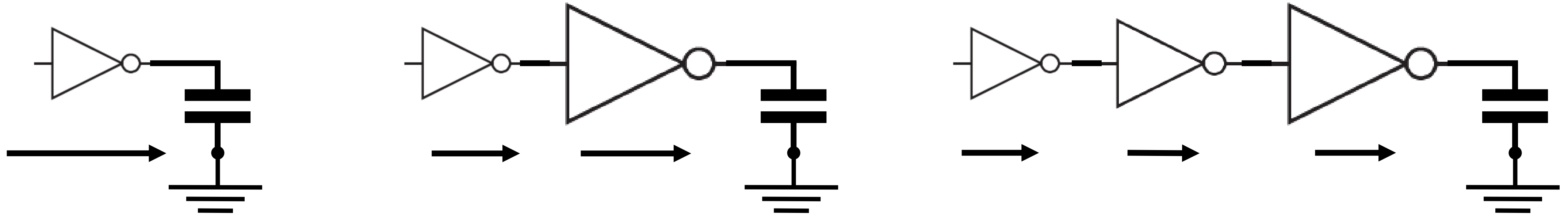
EE-429
Fundamentals of VLSI Design

Logical Effort – Modelling and Analysis

Andreas Burg

Modelling Delay and Sizing Gates

- **CMOS logic designers often face the problem of**
  - Sizing gates in multiple stages of logic to minimize delay
  - Determining the optimum number of logic stages (especially for driving large loads)
  - Rapidly comparing circuit alternatives with optimal sizing
- **Difficulty lies in the impact of sizing one gate to reduce delay on the delay of the preceding gate**
  - Up-sizing one gate to reduce delay increases load (delay) on previous gate
  - Down-sizing one gate increase its delay, but also decreases the load (delay) of the previous gate
- **Example:** driving a large load starting from a small inverter



The Logical Effort model

- **Logical effort is a method to** quickly **analyze and optimize** logic **delay**
  - Uses a simple **linear delay model that includes**
    - the **impact of load on delay**
    - the **impact of sizing on** the **load/delay** of the previous gate
  - Allows back-of-the-envelope calculations
  - Helps make rapid comparisons between alternatives
- **The method is useful for**
  - **Digital circuit designers** trying to optimize their (high-speed) designs by hand
  - **CAD tool developers** to incorporate the key ideas into their tools for design automation

- **Developed in 1999 by Sutherland and Sproull:**
  I. Sutherland, B. Sproull, and D. Harris, Logical Effort: Designing Fast CMOS Circuits, San Francisco, CA: Morgan Kaufmann, 1999.
- See: **http://bibl.ica.jku.at/dc/build/html/logicaleffort/logicaleffort.html**

Technology Independent Delay Model

- **Basic idea: decomposition of the delay**
  - **Decompose the** normalized **delay** of a gate **into** two components
    - A **constant intrinsic delay (stage effort)** term p that depends only on the gate itself
    - A **load-dependent effort delay** $f(type, \text{load}, \text{size})$

$$d = p(type) + f(\text{type}, \text{load}, \text{size})$$

  - **Decompose the load dependent delay** term $f(\text{load})$ **into a product of two components**
    - The **logical effort** g which covers the complexity of the gate (effort to drive a load: increases with the complexity of the gate)
    - The **electrical effort** h which describes the load and the sizing of the gate

$$f(\text{type}, \text{load}, \text{size}) = g(type) * h(load, size)$$

  - p and f **express** the **two delays** $\tau_{p,f}(\text{gate})$ of a gate **relative to the delay of a minimum size, balanced inverter** $\tau_{p,f}(\text{inv})$ **(reference inverter)**

Normalized delays: $p = \frac{\tau_p(\text{gate})}{\tau_p(\text{inv})}$ and $f = \frac{\tau_f(\text{gate})}{\tau_f(\text{inv})}$

Derivation of the Relative Intrinsic Delay

- **Reminder:** Intrinsic delay of a gate depends on on-resistance and the **intrinsic load capacitance C_{int}^{gate}**

$$\tau_p(\text{gate}) = 0.69 \cdot R_{eq}^{gate} C_{int}^{gate}$$

Balanced (reference) inverter (same driving strength as considered gate)

Unloaded: $\tau_p(\text{inv}) = 0.69 \cdot R_{eq}^{inv} C_{int}^{inv}$

C_{int}^{inv}: inverter intrinsic load

Driving itself: $\tau_f(\text{inv}) = 0.69 \cdot R_{eq}^{inv} C_{IN}^{inv}$

C_{IN}^{inv}: inverter input load

- Normalizing with the delay of the reference inverter

$$p = \frac{R_{eq}^{gate} C_{int}^{gate}}{R_{eq}^{inv} C_{int}^{inv}}$$

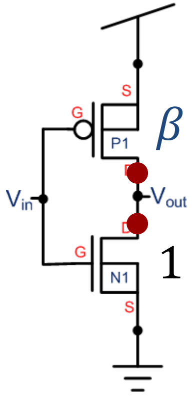
- For a **balanced gate** that is **matched to a reference inverter with the same current drive capability** **($R_{eq}^{gate} = R_{eq}^{inv}$)**

$$p = \frac{C_{int}^{gate}}{C_{int}^{inv}}$$

Model for the Internal Load $C_{int}^{??}$ for Intrinsic Delay

- Internal load is difficult to define precisely, especially since it also depends significantly on the layout. Nevertheless, we need a basic model.

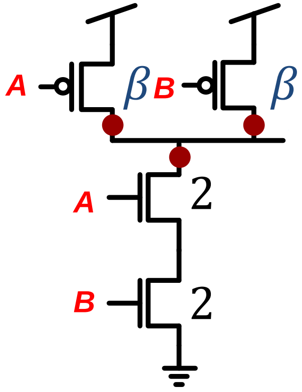
- **Intrinsic load:** dominated by the diffusion capacitance
  - **Relative to the diffusion capacitance of a minimum size nMOS C_0^{diff}**
  - **Depends linearly on the width** of the transistors that contribute to it
  - **Consider only the load** that is **attached directly to the output** node of the gate



$$C_{int}^{inv} = (\beta + 1)C_0^{diff}$$

$$\beta = 2 : C_{int}^{inv} \sim 3\ C_0^{diff}$$

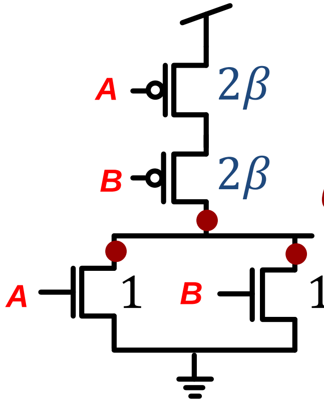
$$p^{inv} = 1$$



$$C_{int}^{nand} = (2\beta + 2)C_0^{diff}$$

$$\beta = 2 : C_{int}^{nand} \sim 6\ C_0^{diff}$$

$$p^{nand} = 2$$



$$C_{int}^{nor} = (2\beta + 2)C_0^{diff}$$

$$\beta = 2 : C_{int}^{nor} \sim 6\ C_0^{diff}$$

$$p^{nor} = 2$$

Split Effort Delay into Logical- and Electrical-Effort

- **Start from the effort delay of a gate**

$$\tau_f(\text{gate}) = 0.69 \cdot R_{eq}^{gate} \cdot C_L$$

Balanced (reference) inverter (same driving strength as considered gate)

Unloaded: $\tau_p(\text{inv}) = 0.69 \cdot R_{eq}^{inv} C_{int}^{inv}$

C_{int}^{inv}: inverter intrinsic load

Driving itself: $\tau_f(\text{inv}) = 0.69 \cdot R_{eq}^{inv} C_{IN}^{inv}$

C_{IN}^{inv}: inverter input load

- **Normalizing with the effort delay of a reference inverter with same current drive capability** $(R_{eq}^{gate} = R_{eq}^{inv})$

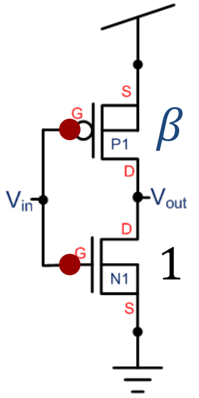
$$f(C_L) = \frac{R_{eq}^{gate} \cdot C_L}{R_{eq}^{inv} \cdot C_{IN}^{inv}} * \underbrace{\frac{C_{IN}^{gate}}{C_{IN}^{gate}}}_{1} = \frac{C_L}{C_{IN}^{inv}} * \underbrace{\frac{C_{IN}^{gate}}{C_{IN}^{gate}}}_{1} = \underbrace{\frac{C_{IN}^{gate}}{C_{IN}^{inv}}}_{g} * \underbrace{\frac{C_L}{C_{IN}^{gate}}}_{h}$$

Logical effort — Electrical effort

- **Logical effort**: complexity of the gate represented by the additional input load compared to an equally strong inverter
- **Electrical effort**: difficulty to drive the output relative to the difficulty of the previous gate to drive the input

Deriving the Input Capacitance/Load

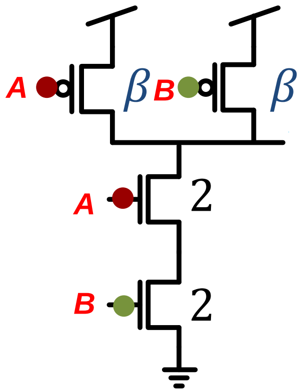
- **Input capacitance:** load that the input of a gate presents to the preceding gate
  - **Relative to gate capacitance of a minimum size nMOS** C_0
  - **Depends linearly on the width** of the transistors that contribute to it
  - Value of C_0 is irrelevant when
    - used to calculate $g = C_{IN}^{gate}/C_{IN}^{inv}$ (ratio of gate capacitances)
    - when the C_{L} is only the input of other gates since then $C_{\mathrm{L}} \sim C_0$ and $h = C_{\mathrm{L}}/C_{IN}^{inv}$ independent of C_0



$C_{IN}^{inv} = (\beta + 1)C_0$

$\beta = 2 : C_{IN}^{inv} \sim 3C_0$

$g = 1$



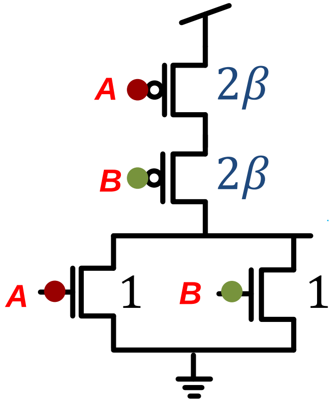
$C_{IN}^{nand}(A) = (\beta + 2)C_0$

$\beta = 2 : C_{IN}^{nand}(A) \sim 4C_0$

$C_{IN}^{nand}(B) = (\beta + 2)C_0$

$\beta = 2 : C_{IN}^{nand}(B) \sim 4C_0$

$g = 4/3$



$C_{IN}^{nor}(A) = (2\beta + 1)C_0$

$\beta = 2 : C_{IN}^{nor}(A) \sim 5C_0$

$C_{IN}^{nor}(B) = (2\beta + 1)C_0$

$\beta = 2 : C_{IN}^{nor}(B) \sim 5C_0$

$g = 5/3$

Logical Effort Gate Delay Summary

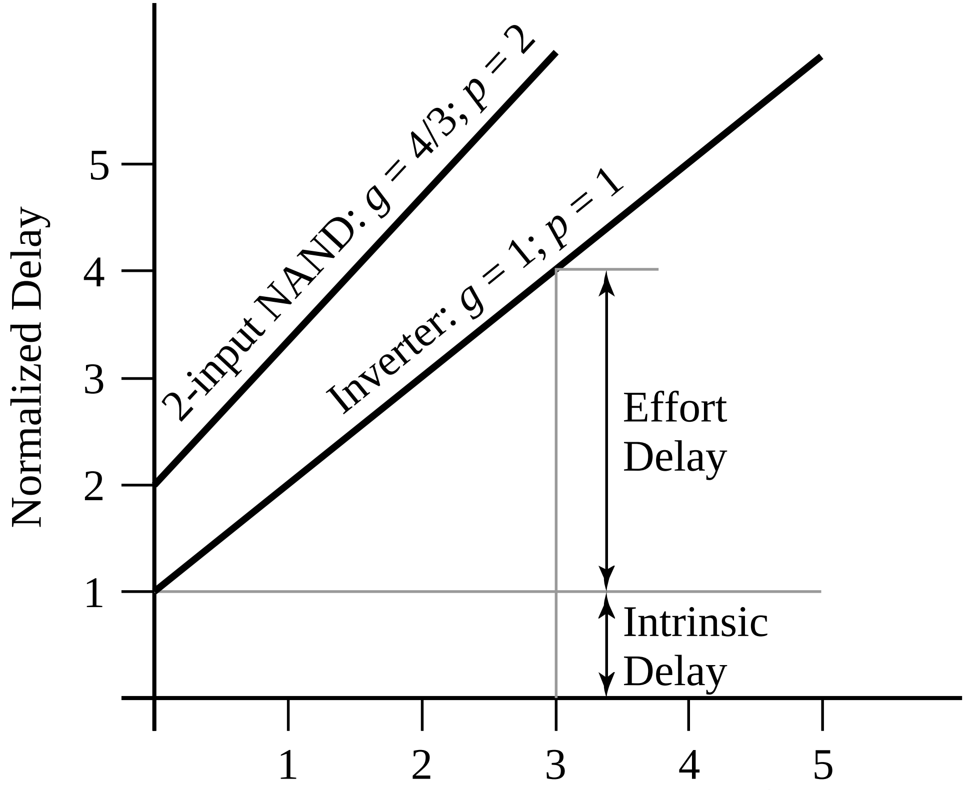
- **Gate delay is modelled as**

$$d = p + g * h$$

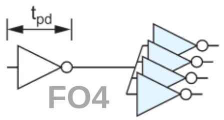
  - **Intrinsic delay p:** load independent delay
    - Depends only on the gate
    - Approximated by the number of unit diffusion capacitances (C_0^{diff}) connected to the output of the gate, relative to an equivalent inverter $$p = \frac{C_{int}^{gate}}{C_{int}^{inv}}$$
  - **Logical effort g:** cost for driving an output load (due to the complexity of the gate)
    - Depends only on the gate (not on the load), but can be different for different inputs of the gate
    - Given by the ratio of capacitances at the input of a gate compared to the capacitances of an equivalent inverter $$g = \frac{C_{IN}^{gate}}{C_{IN}^{inv}}$$
  - **Electrical effort h:** size of the load relative to the load the gate to a previous gate
    - Depends on the gate and the load
    - Given by the ratio of the load to the input capacitance of the gate $$h = \frac{C_{\mathrm{L}}}{C_{IN}^{gate}}$$

Logical Effort Gate Delay Examples

- **Consider two examples: INV & NAND**
  - NAND has a larger intrinsic delay
  - Drive capability of NAND is the same as for the inverter, hence, **delay increase with higher absolute load should be the same**
    - "Normalized delay" reflects how well the gate with the same input capacitance as an inverter can drive a load
  - However, we consider the load relative to the input capacitance of the gate (**FANOUT**):
    - This reflects the impact of facilitating drive through up-sizing on the previous gate
    - Driving a larger relative load, is more "difficult" and leads to larger delay
  - Delay increase with higher relative (to the input) load is more



FANOUT
Electrical effort
$h = C_{out}/C_{in}$



Logical Effort for Popular Gates

- **Intrinsic delay** and **logical effort** **for frequently used gates with $\beta = 2$:**
  - **Note that** Intrinsic delay and logical effort are independent of the gate sizing, but they depend on the PMOS/NMOS ratio (β)

intrinsic delay

Gate Type	Number of Inputs				
	1	2	3	4	n
inverter	1				
NAND		2	3	4	n
NOR		2	3	4	n
tristate, multiplexer	2	4	6	8	$2n$

logical effort

Gate Type	Number of Inputs				
	1	2	3	4	n
inverter	1				
NAND		4/3	5/3	6/3	$(n+2)/3$
NOR		5/3	7/3	9/3	$(2n+1)/3$
tristate, multiplexer	2	2	2	2	2
XOR, XNOR		4, 4	6, 12, 6	8, 16, 16, 8	

Difference between NAND and NOR becomes apparent in logical effort

Extracting Delay Constant from Ring Oscillator

- **Relating the logical effort "relative" delay to an absolute delay requires two delay constants:** $\tau_p(\text{inv})$ **and** $\tau_f(\text{inv})$
- **Delay constants are related by their unit loads (diffusion load vs. gate load)**

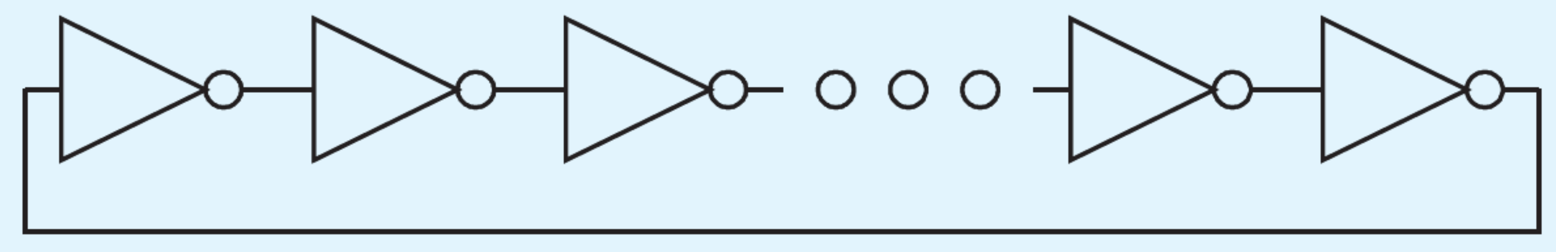
$$\tau_p(\text{inv})/\tau_f(\text{inv}) = C_0^{diff}/C_0$$

  - **For many** modern **processes**, we find that $C_0^{diff} \approx C_0$ ➔ $\boldsymbol{\tau_p(inv) = \tau_f(inv) = \tau(inv)}$

- **Experimental setup** to extract $\tau(inv)$: ***N*-stage ring oscillator**

Logical effort delay

$$p_{inv} = 1$$
$$g_{inv} = 1$$
$$h = 1$$
$$d_{osc} = N \cdot p_{inv} + N \cdot g_{inv} \cdot h = 2N$$



Measured oscillator frequency $f_{osc} = \frac{1}{2 \cdot 2N \cdot \tau(inv)}$ ➔ $\boldsymbol{\tau(inv) = 4Nf_{osc}}$

Path Delay Calculation with Logical Effort

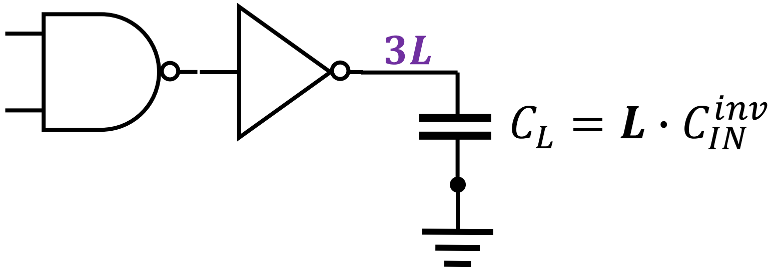
- **Logical effort** model allows **to** easily **calculate path delays**
  - Normalized **path delay obtained by summing** of normalized **gate delays**

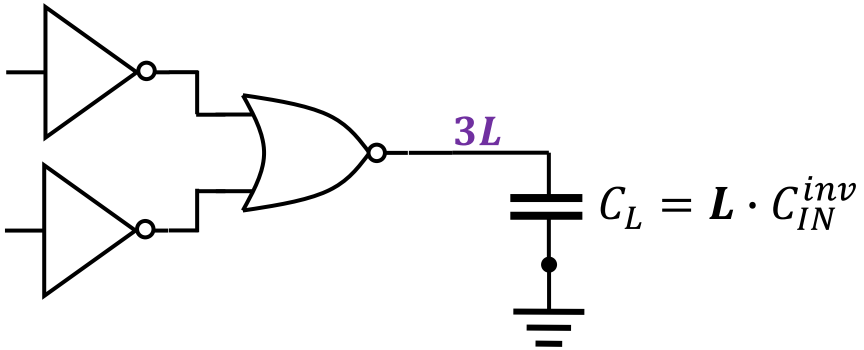
- **Procedure for path delay calculation**

1. For each gate i **calculate input capacitance C_{IN}^{i}** and **output load C_{L}^{i}** relative to min. inverter
   - Without branching (drives only one next gate… see later): $C_L^i = C_{IN}^{i+1}$
   - Consider that **C_{IN}^{i} scales with size γ** compared to the minimum size gate
2. **Evaluate intrinsic delay** p, **logical effort** g, **and electrical effort** h for **each gate** in the path
3. **Compute gate delays and sum** up to obtain the path delay

Path Delay Calculation Examples

- **Example 1:** compare two AND gate implementations (min. size, $\beta = 2$)





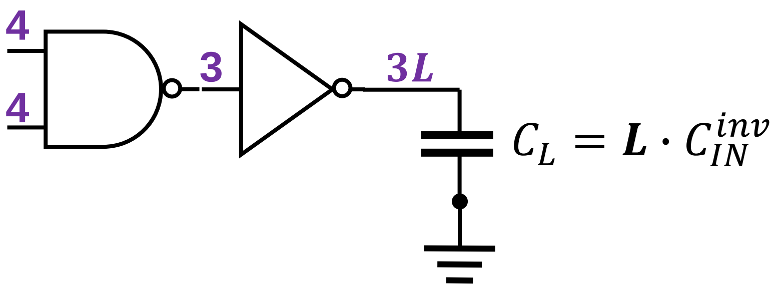
	i = 1	i = 2
p		
g		
h		

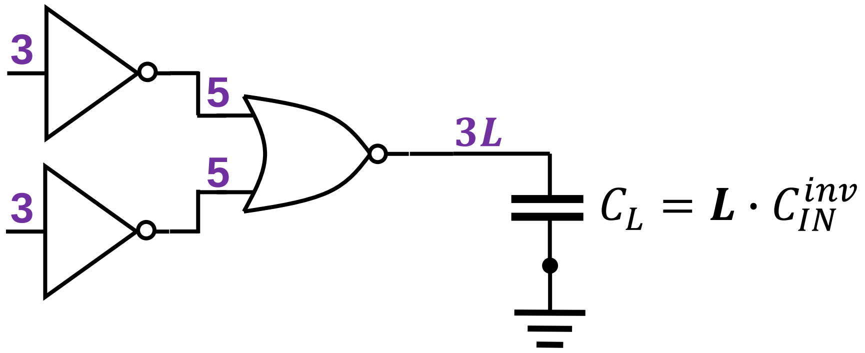
	i = 1	i = 2
p		
g		
h		

DELAY:

Path Delay Calculation Examples

- **Example 1:** compare two AND gate implementations (min. size, $\beta = 2$)





	i = 1	i = 2
p	2	1
g	4/3	1
h	3/4	$3L/3$

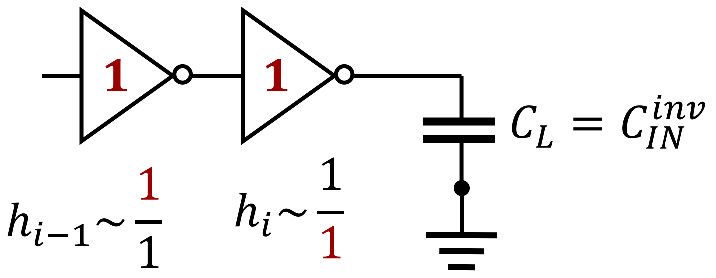
	i = 1	i = 2
p	1	2
g	1	5/3
h	5/3	$3L/5$

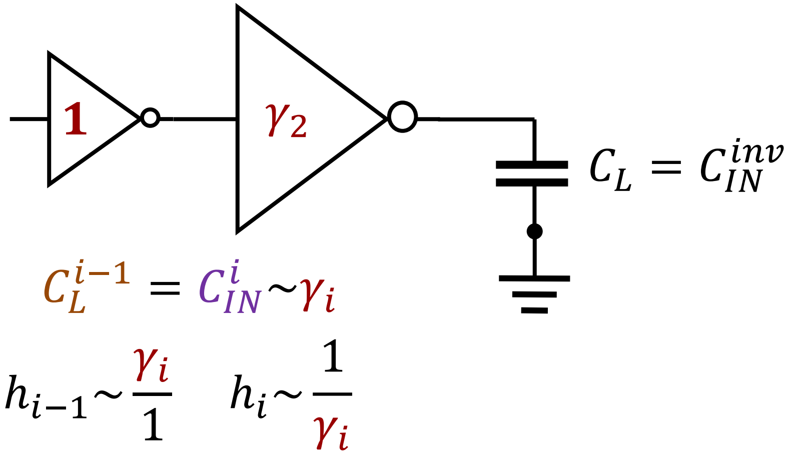
DELAY: $4 + L$ $\quad \frac{14}{3} + L$

Gate Sizing

- **How to adjust the size of gates in a path to minimize delay?**
  - **Up-Sizing factors γ_i:** specifies size of gate i relative to the corresponding minimum size gate
  - Assume that the **size of the first gate** in a chain **is given** (e.g., $\boldsymbol{\gamma_1 \equiv 1}$) **and can not be altered**

- **Implications of up-sizing a gate on a path**



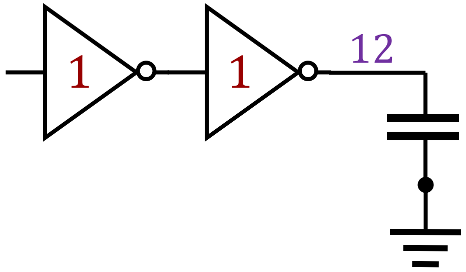


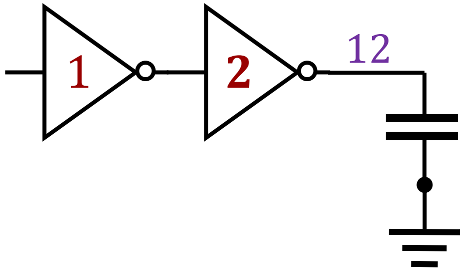
  - Delay (electrical effort h) of the up-sized gate reduces
  - Delay (electrical effort h) of the previous gate increases

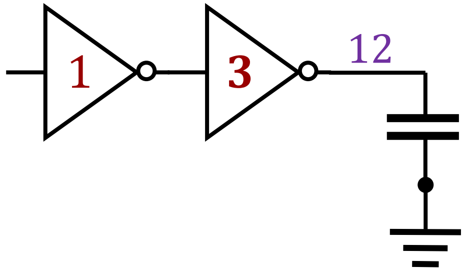
- **Local optimization is not sufficient:** need to capture impact of sizing on full path

Impact of Gate Sizing Example

- **Two inverters driving a load of** $C_L = \mathbf{4} \cdot C_{IN}^{inv}$ $(\beta = 2)$
  - Consider three configurations







	i = 1	i = 2
p		
g		
h		

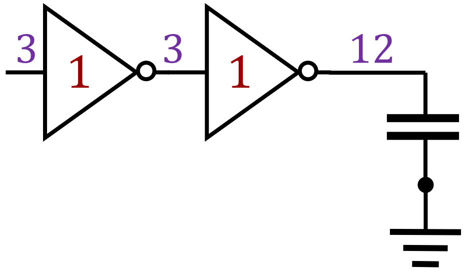
	i = 1	i = 2
p		
g		
h		

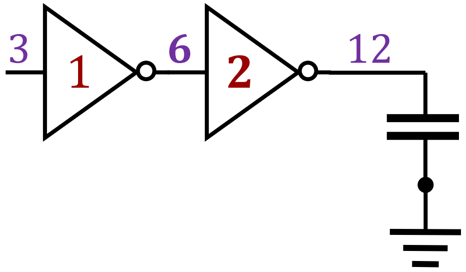
	i = 1	i = 2
p		
g		
h		

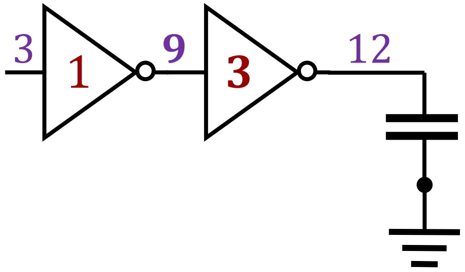
DELAY:

Impact of Gate Sizing Example

- **Two inverters driving a load of** $C_L = \mathbf{4} \cdot C_{IN}^{inv}$ $(\beta = 2)$
  - Consider three configurations







	i = 1	i = 2
p	1	1
g	1	1
h	3/3	12/3

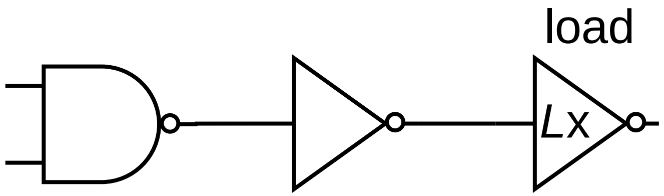
	i = 1	i = 2
p	1	1
g	1	1
h	6/3	12/6

	i = 1	i = 2
p	1	1
g	1	1
h	9/3	12/9

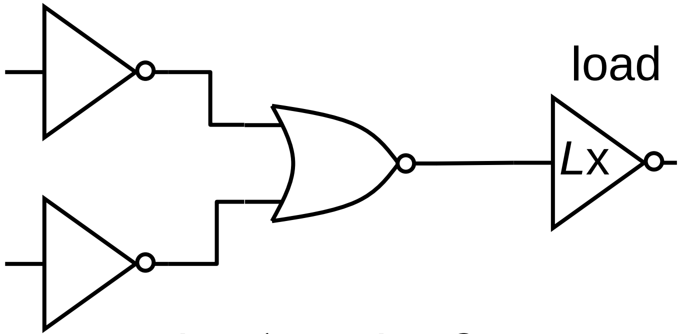
DELAY: 7 **6** 6.33

Topology Analysis

- Consider the following **two options to implement the same logic function:** $Z = \overline{\overline{A \cdot B}}$ that drives a large load (Lx INV)
  - Which option is better (potentially as a function of L)?



	i = 1	i = 2
p		
g		
h		

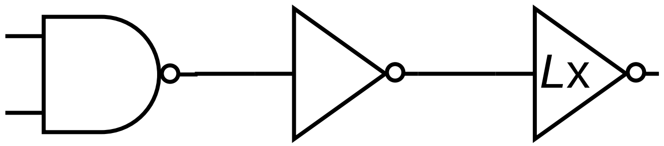


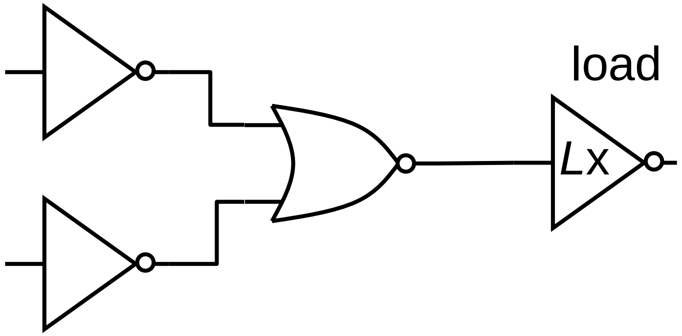
	i = 1	i = 2
p		
g		
h		

DELAY:

Topology Analysis

- Consider the following **two options to implement the same logic function:** $Z = \overline{\overline{A \cdot B}}$ that drives a large load (Lx INV)
  - Which option is better (potentially as a function of L)?





	i = 1	i = 2
p	2	1
g	4/3	1
h	3/4	3L/3

	i = 1	i = 2
p	1	2
g	1	5/3
h	5/3	3L/5

DELAY: 4 + L   4.66 + L

Always better

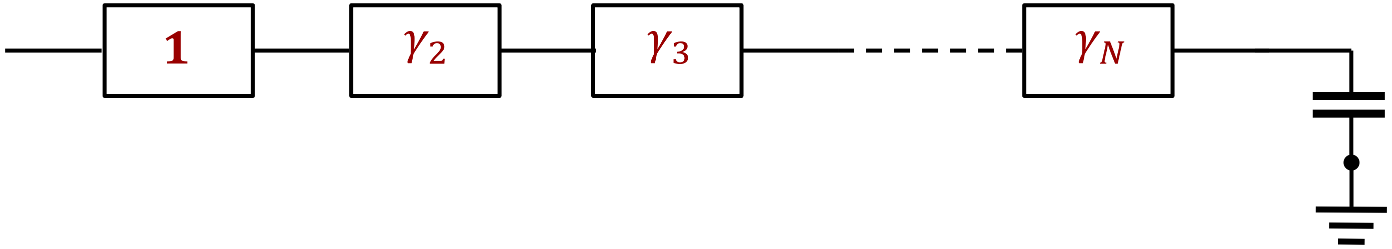
EE-429
Fundamentals of VLSI Design

Logical Effort – Optimization

Andreas Burg

Optimum Sizing with Logical Effort (Derivation)

- Consider the schematic and the delay of a generic path



		i = 1	i = 2	i = 3	i = N
Fixed: given by circuit topology	**p**	$\boldsymbol{p_1}$	$\boldsymbol{p_2}$	$\boldsymbol{p_3}$	$\boldsymbol{p_N}$
	g	$\boldsymbol{g_1}$	$\boldsymbol{g_2}$	$\boldsymbol{g_3}$	$\boldsymbol{g_N}$
unknown	**h**	h_1	h_2	h_3	h_N

$h_N \longrightarrow h_i \sim \frac{\gamma_{i+1}}{\gamma_i} \longrightarrow$ **unknown**

$$\mathrm{D} = \boldsymbol{p_1} + \underbrace{g_1 h_1}_{f_1} + \boldsymbol{p_2} + \underbrace{g_2 h_2}_{f_2} + \boldsymbol{p_3} + \underbrace{g_3 h_3}_{f_3} + \quad + \boldsymbol{p_N} + \underbrace{g_N h_N}_{f_N}$$

- Minimize delay ⇒ **minimize** $\boldsymbol{f_1 + f_2 + \cdots + f_N = \sum_{i=1}^{N} f_i}$ (path effort **delay**) ⟶ **DIFFICULT!**

Optimum Sizing with Logical Effort (Derivation)

- How to **minimize** the path effort **delay** $\sum_{i=1}^{N} f_i$

- **Trick:** consider the **path effort**

$$F = f_1 \cdot f_2 \cdot \cdots \cdot f_N = \prod_{i=1}^{N} f_i = \underbrace{\prod_{i=1}^{N} g_i}_{G} \cdot \underbrace{\prod_{i=1}^{N} h_i}_{H} = G \cdot H$$

G is known from the topology

  - **Computing** $H = \underbrace{\frac{\gamma_2 C_0^2}{\gamma_1 C_0^1}}_{h_1} \cdot \underbrace{\frac{\gamma_3 C_0^3}{\gamma_2 C_0^2}}_{h_2} \cdot \underbrace{\frac{\gamma_4 C_0^4}{\gamma_3 C_0^3}}_{h_3} \cdot \cdots \cdot \underbrace{\frac{C_L}{\gamma_N C_0^N}}_{h_N} = \frac{C_L}{\gamma_1 C_0^1}$ → **Independent of internal sizes**

    as geometric mean is possible **without knowledge of** $\gamma_2 \ldots \gamma_N$**!**

    - ***H* depends only on the first gate and the last load of the path**

  - Exploit ability to compute H with **theorem of arithmetic mean & geometric mean**

C_0^i : input capacitance of gate i with minimum size

$$\frac{1}{N}\sum_{i=1}^{N} f_i \geq \sqrt[N]{F}$$

Equality if $f_1 = f_2 = \cdots = f_N$

Equality minimizes $f_1 + f_2 + \cdots + f_N$

Delay with Optimum Sizing with Logical Effort

- **Effort delay is minimized if**

$$\hat{f} = f_1 = f_2 = \cdots f_N = \sqrt[N]{\boldsymbol{F}} = \sqrt[N]{\boldsymbol{G} \cdot \boldsymbol{H}}$$

$$\boldsymbol{G} = \prod_{i=1}^{N} \boldsymbol{g_i} \qquad \mathbf{H} = \prod_{i=1}^{N} \boldsymbol{h_i} = \frac{\boldsymbol{C_L}}{\boldsymbol{C_0^1 \gamma_1}}$$

  - $\boldsymbol{G}$: from gate types given by the path topology
  - $\boldsymbol{C_L}$: load given by the problem statement (in same unit as C_0^1)
  - $\boldsymbol{C_0^1}$: minimum size input capacitance of first gate type (in same unit as C_L)
  - $\boldsymbol{\gamma_1}$: given relative sizing of the first gate

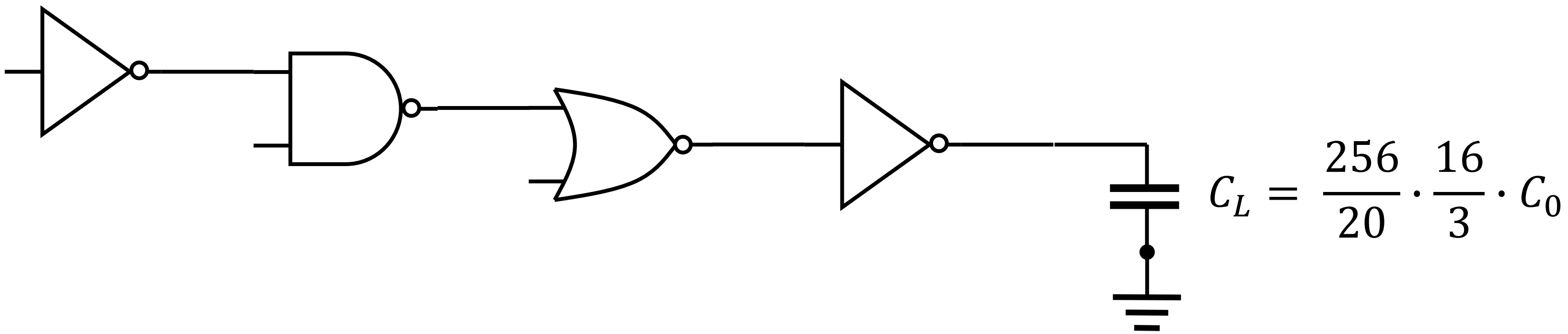
- **Minimum achievable path delay**
  - Can be derived without knowledge of the corresponding gate sizes!

p_i: given, $\quad g_i \cdot h_i = \sqrt[N]{\frac{C_L}{C_0^1 \gamma_1} \cdot \prod_{i=1}^{N} g_i} = \hat{f}$

Total path delay with optimum sizing = $\sum_{i=1}^{N} p_i + N \cdot \sqrt[N]{\frac{C_L}{C_0^1 \gamma_1} \cdot \prod_{i=1}^{N} g_i}$

Calculating Delay with Optimum Sizing (Example)

- **Consider the following path** $(\beta = 2)$ **:**



	i = 1	i = 2	i = 3	i = 4
p				
g				

$N =$

$H =$

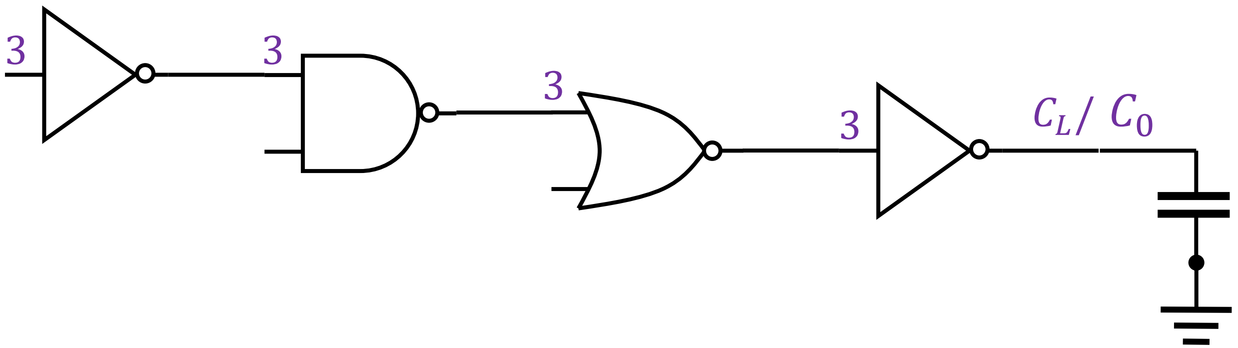
$G =$

$\hat{f} =$

$D_{min} =$

Calculating Delay with Optimum Sizing (Example)

- **Consider the following path** $(\beta = 2)$ **:**



$$C_L = \frac{256}{20} \cdot \frac{16}{3} \cdot C_0$$

	i = 1	i = 2	i = 3	i = 4
p	1	2	2	1
g	1	4/3	5/3	1

$$N = 4 \qquad H = \frac{\frac{256}{20} \cdot \frac{16}{3}}{3} \qquad G = 1 \cdot \frac{4}{3} \cdot \frac{5}{3} \cdot 1 = \frac{20}{9} \qquad \hat{f} = \sqrt[4]{HG} = \frac{8}{3}$$

$$D_{min} = \sum p_i + N \cdot \hat{f} = 6 + \frac{32}{3} \approx 16.66$$

Deriving Optimum Gate Sizes with Logical Effort

- **Optimum gate sizes $\boldsymbol{\gamma_i}$** can be **derived from** knowledge of $\boldsymbol{\hat{f}} = g_i \cdot h_i$
  - h_i depends only on γ_i and γ_{i+1} OR on γ_i and C_L
  - γ_1 and C_L are known

$$\hat{f} = \sqrt[N]{\frac{C_L}{C_0^1 \gamma_1} \cdot \prod_{i=1}^{N} g_i}$$

$$\gamma_1 = 1$$

- **Two strategies:**
  - Input → Output (left to right)

$$\frac{C_0^i}{C_0^j} = \frac{g_i}{g_j}$$

$$f_1 = \hat{f} = g_1 \cdot \frac{\gamma_2}{\gamma_1} \cdot \frac{C_0^2}{C_0^1} = \frac{\gamma_2}{\gamma_1} \cdot g_2 \quad \longmapsto \quad \gamma_2 = \hat{f} \cdot \frac{\gamma_1}{g_2}$$

$$f_2 = \hat{f} = g_2 \cdot \frac{\gamma_3}{\gamma_2} \cdot \frac{C_0^3}{C_0^2} = \frac{\gamma_3}{\gamma_2} \cdot g_3 \quad \longmapsto \quad \gamma_3 = \hat{f} \cdot \frac{\gamma_2}{g_3} = \hat{f}^2 \cdot \frac{\gamma_1}{g_2 g_3}$$

$$f_3 = \hat{f} = g_3 \cdot \frac{\gamma_4}{\gamma_3} \cdot \frac{C_0^4}{C_0^3} = \frac{\gamma_4}{\gamma_3} \cdot g_4 \quad \longmapsto \quad \gamma_4 = \hat{f} \cdot \frac{\gamma_3}{g_4} = \hat{f}^3 \cdot \frac{\gamma_1}{g_2 g_3 g_4}$$

$$f_n = \ldots \quad \longmapsto \quad \gamma_n = \hat{f}^{n-1} \cdot \gamma_1 \cdot \prod_{i=2}^{n} g_i^{-1}$$

Deriving Optimum Gate Sizes with Logical Effort

- Output → Input (right to left)

$$f_N = \hat{f} = g_N \cdot \frac{1}{\gamma_N} \cdot \frac{C_L}{C_0^N} \quad \mapsto \quad \gamma_N = \frac{1}{\hat{f}} \frac{C_L}{C_0^N} \cdot g_N$$

$$f_{N-1} = \hat{f} = g_{N-1} \cdot \frac{\gamma_N}{\gamma_{N-1}} \cdot \frac{C_0^N}{C_0^{N-1}} \quad \mapsto \quad \gamma_{N-1} = \frac{1}{\hat{f}} \cdot g_{N-1} \cdot \gamma_N \cdot \frac{C_0^N}{C_0^{N-1}} = \frac{1}{\hat{f}^2} \cdot \frac{C_L}{C_0^{N-1}} \cdot g_{N-1} \cdot g_N$$

$$f_{N-2} = \hat{f} = g_{N-2} \cdot \frac{\gamma_{N-1}}{\gamma_{N-2}} \cdot \frac{C_0^{N-1}}{C_0^{N-2}} \quad \mapsto \quad \gamma_{N-2} = \frac{1}{\hat{f}} \cdot g_{N-2} \cdot \gamma_N \cdot \frac{C_0^{N-1}}{C_0^{N-2}} = \frac{1}{\hat{f}^3} \cdot \frac{C_L}{C_0^{N-2}} \cdot g_{N-2} \cdot g_{N-1} \cdot g_N$$

$$f_n = \ldots \quad \mapsto \quad \gamma_n = \frac{1}{\hat{f}^{N-n+1}} \cdot \frac{C_L}{C_0^n} \cdot \prod_{i=n}^{N} g_i$$

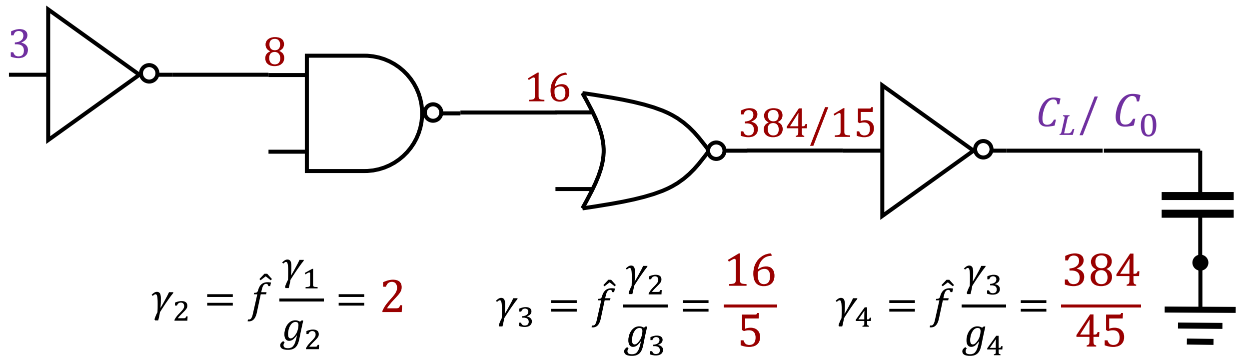
- Drive strength scales exponentially with every stage by a factor of $\hat{f}$, corrected by the logical effort of the stage

$$\frac{\gamma_n}{\gamma_{n-1}} = \hat{f}/g_n$$

Calculating Delay with Optimum Sizing (Example)

- **Consider the following path** $(\beta = 2)$ **:**

$$D_{min} = \sum p_i + N \cdot \hat{f} = 6 + \frac{32}{3} \approx 16$$



$$C_L = \frac{256}{20} \cdot \frac{16}{3} \cdot C_0$$

$$\gamma_2 = \hat{f}\frac{\gamma_1}{g_2} = 2 \qquad \gamma_3 = \hat{f}\frac{\gamma_2}{g_3} = \frac{16}{5} \qquad \gamma_4 = \hat{f}\frac{\gamma_3}{g_4} = \frac{384}{45}$$

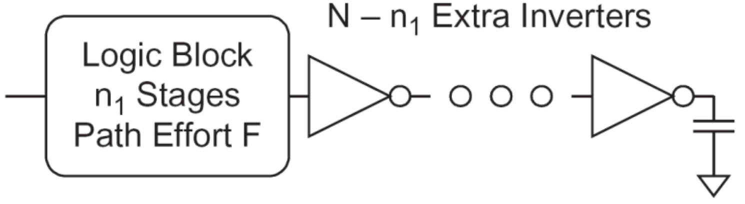
	i = 1	i = 2	i = 3	i = 4
p	1	2	2	1
g	1	4/3	5/3	1
h	8/3	16/8	24/15	8/3

$$\hat{f} = \sqrt[4]{HG} = \frac{8}{3}$$

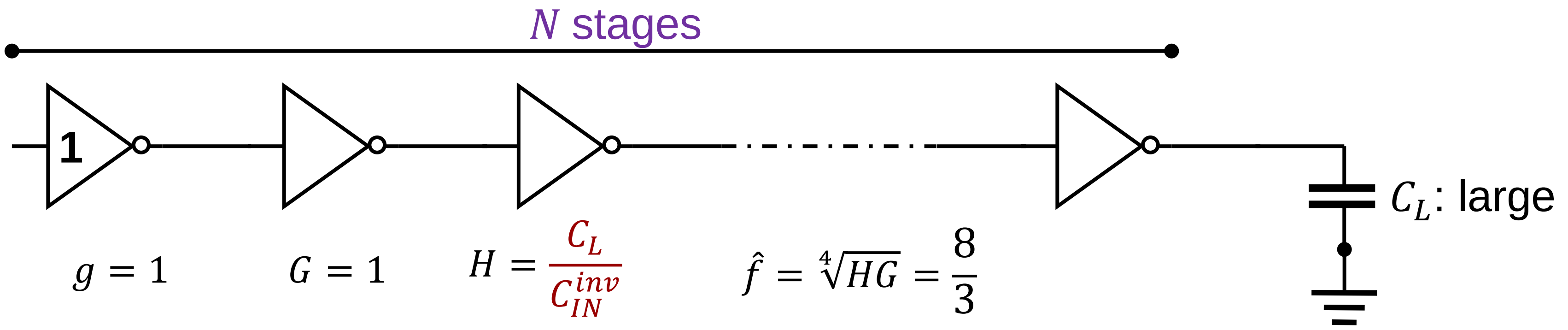
$$D = \sum p_i + \sum g_i h = 6 + \frac{8}{3} + \frac{4}{3}\frac{16}{8} + \frac{5}{3}\frac{24}{45} + \frac{8}{3} \approx 16.66 = D_{min}$$

Driving Large Loads

- We often face the **challenge to drive a large load** starting **from a small gate**
  - Single large buffer is not efficient as it puts too much load on the small initial gate



- **Solution**: chain of multiple buffers with increasing size
  - More buffers, more intrinsic delay, but less effort delay…



$g = 1$ $\quad G = 1$ $\quad H = \frac{C_L}{C_{IN}^{inv}}$ $\quad \hat{f} = \sqrt[4]{HG} = \frac{8}{3}$

  - How to size those inverters? $\gamma_n = \hat{f}^{n-1} = \left(\frac{C_L}{C_{IN}^{inv}}\right)^{\frac{n-1}{N}}$ ⟶ **geometric sizing**

$$D_{min}(N) = N + N\sqrt[N]{C_L/C_{IN}^{inv}}$$

Driving Large Loads

- **How many buffers/inverter stages should we use for min. delay?**
  - Find the minimum of $D_{min}(N)$ with respect to N

$$\frac{\partial D_{min}(N)}{\partial N} = -H^{\frac{1}{N}} \ln H^{\frac{1}{N}} + H^{\frac{1}{N}} + 1 = 0$$

  - Difficult to solve for N!

$$D_{min}(N) = N + N\sqrt[N]{H}$$
$$H^{\frac{1}{N}} = \hat{f}$$

- **Rewrite to find stage effort $\rho = H^{\frac{1}{N}}$ that minimizes $D_{min}(N)$**

$$-\rho \ln \rho + \rho + 1 = \rho(1 - \ln \rho) = 1$$

  - Numerical evaluation leads to $\rho_{opt} \approx 3.69$ (optimum stage effort) → **We often use FO4 as a reference since is close to the optimum stage effort**

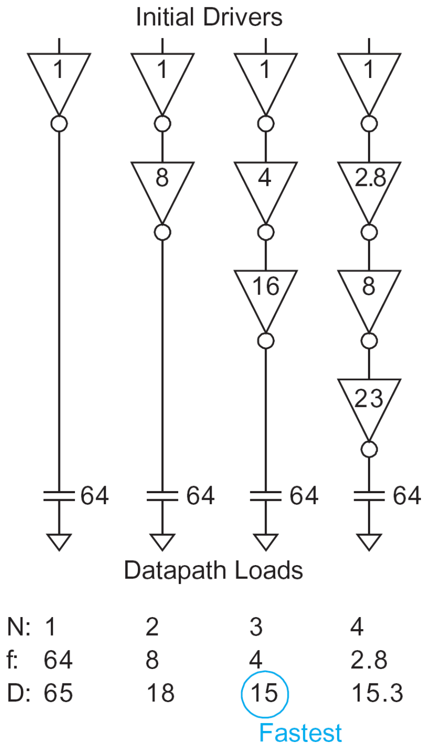
- **Reconstruct the optimum number of stages from ρ_{opt} and H**

$$\hat{N} = 1/\log_H(\rho_{opt})$$

Driving Large Loads

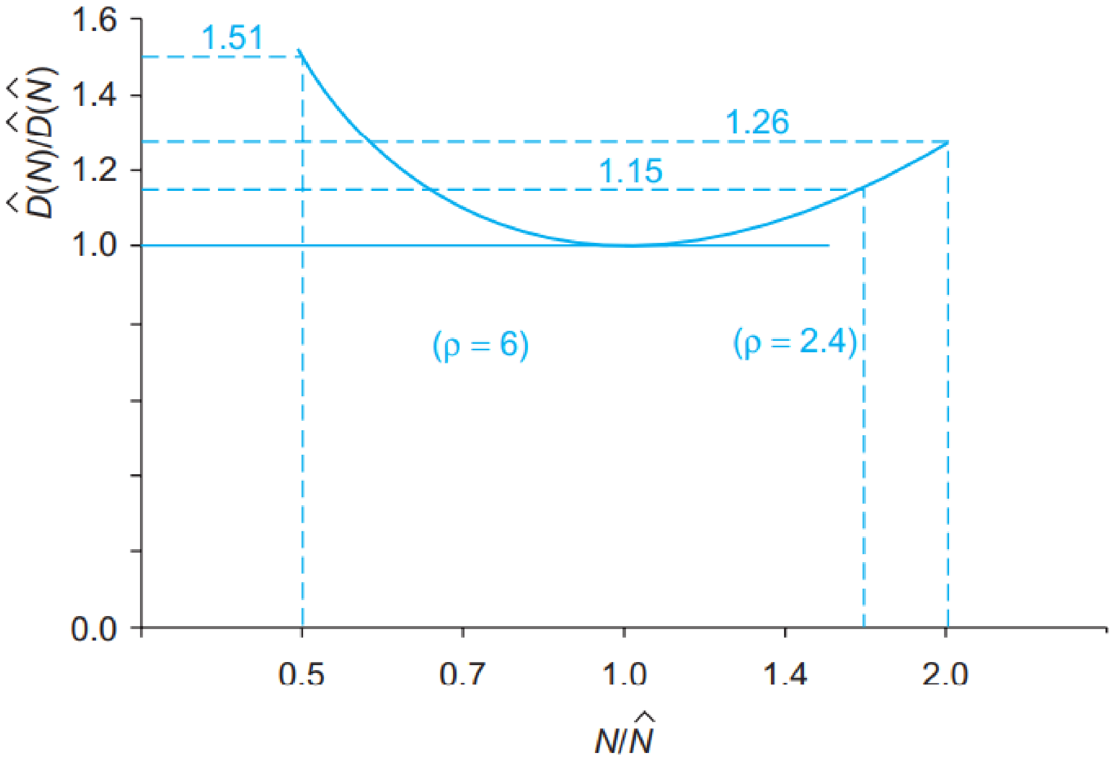
- **Engineering solution: smart trial and error**
  - Useful when for example $p > 1$
  - Use that **only integer choices** are allowed **for N**
  - For inverters, may even restrict N to be even

  - A simple procedure with binary search
    1. Start with $N = 1$
    2. Compute $D_{min}(N)$ and $\partial D_{min}(N)/\partial N$
    3. As long as $\frac{\partial D_{min}(N)}{\partial N} < 0$ double N and proceed with 2.
       Once $\frac{\partial D_{min}(N)}{\partial N} > 0$, proceed with a binary search targeting $\frac{\partial D_{min}(N)}{\partial N} = 0$



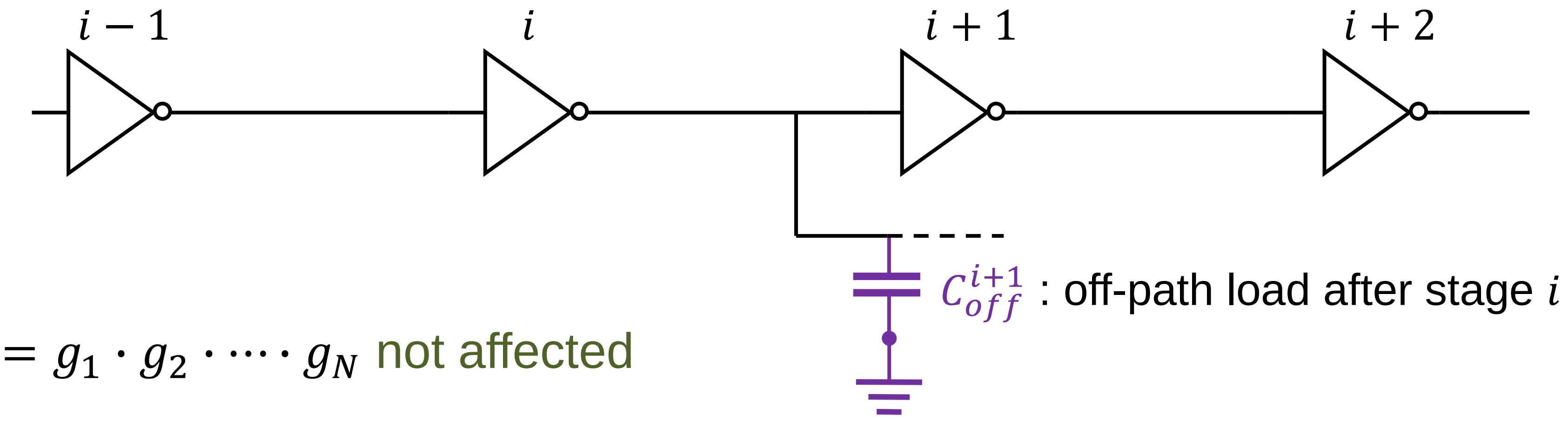
Driving Large Loads

- **Finding the** exact optimal solution for the **best number of stages is tricky** and has many approximations
- **How sensitive is the delay to small errors in the solution?**
- Consider the delay as a function of the deviation from the optimal number of stages:
  - **Sensitivity of the delay to variations in the number of stages is low**



Dealing with Branches

- **Digital Circuits** are rarely linear, but **typically branch out** (tree-like, with FO>1)



  - **Path logical effort**: $G = g_1 \cdot g_2 \cdot \cdots \cdot g_N$ not affected

  - **Path electrical effort:** $H = h_1 \cdot \cdots \cdot h_{i-1} \cdot \underbrace{\frac{\gamma_{i+1} C_0^{i+1} + C_{off}^{i+1}}{\gamma_i C_0^i}}_{\neq h_{i+1} = \frac{\gamma_{i+1} C_0^{i+1}}{\gamma_i C_0^i}} \cdot h_{i+1} \ldots \cdot h_N \neq \frac{C_L}{C_0^1}$ **is affected**

Dealing with Branches using Branch Efforts

- **Basic idea:** for consistency, **move branching overhead into separate factor**
  - Keep same definition of $H = \frac{C_L}{C_0^1}$ also with branches
  - Define a per-stage branch effort c so that $b_i \cdot \underbrace{\frac{\gamma_{i+1} C_0^{i+1}}{\gamma_i C_0^i}}_{h_i} = \frac{\gamma_{i+1} C_0^{i+1} + C_{off}^{i+1}}{\gamma_i C_0^i}$

- **Path logical effort** with branches:

$$G = \prod_{i=1}^{N} g_i \qquad \mathrm{H} = \prod_{i=1}^{N} h_i = \frac{C_L}{C_0^1 \gamma_1} \qquad B = \prod_{i=1}^{N} b_i$$

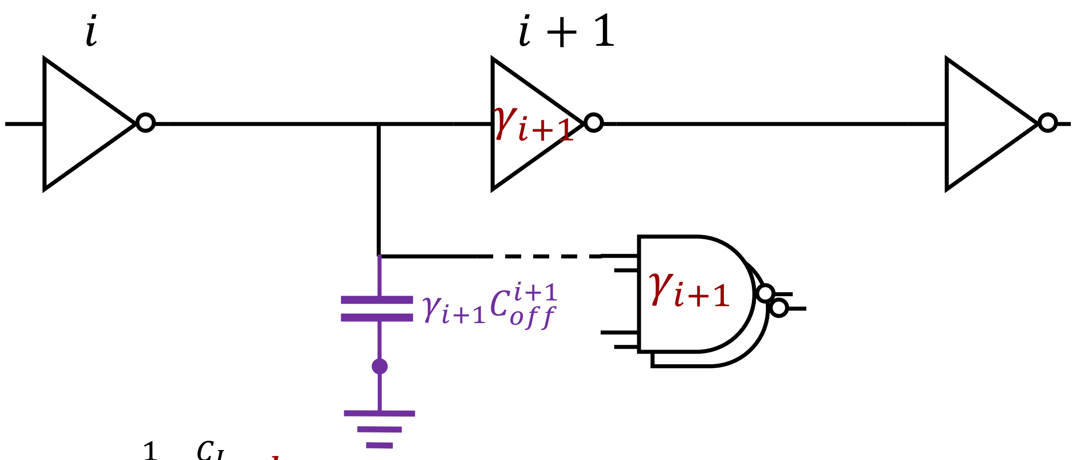
Path effort: $F = GBH$ Stage effort: $\hat{f} = \sqrt[N]{GBH}$

Sizing with Branches

- The **path branch effort B depends on the sizing γ_i** at the branches
  - **Calculation of $\hat{f} = \sqrt[N]{GB(\gamma)H}$ prior to sizing is no longer possible**

- **Trick** to re-enable sizing: **perform same sizing on all paths on branch**
  - Branch effort becomes again independent of γ_{i+1}

$$b_i = \frac{\gamma_{i+1}C_0^{i+1} + \gamma_{i+1}C_{off}^{i+1}}{\gamma_{i+1}C_0^{i+1}} = \frac{C_0^{i+1} + C_{off}^{i+1}}{C_0^{i+1}}$$



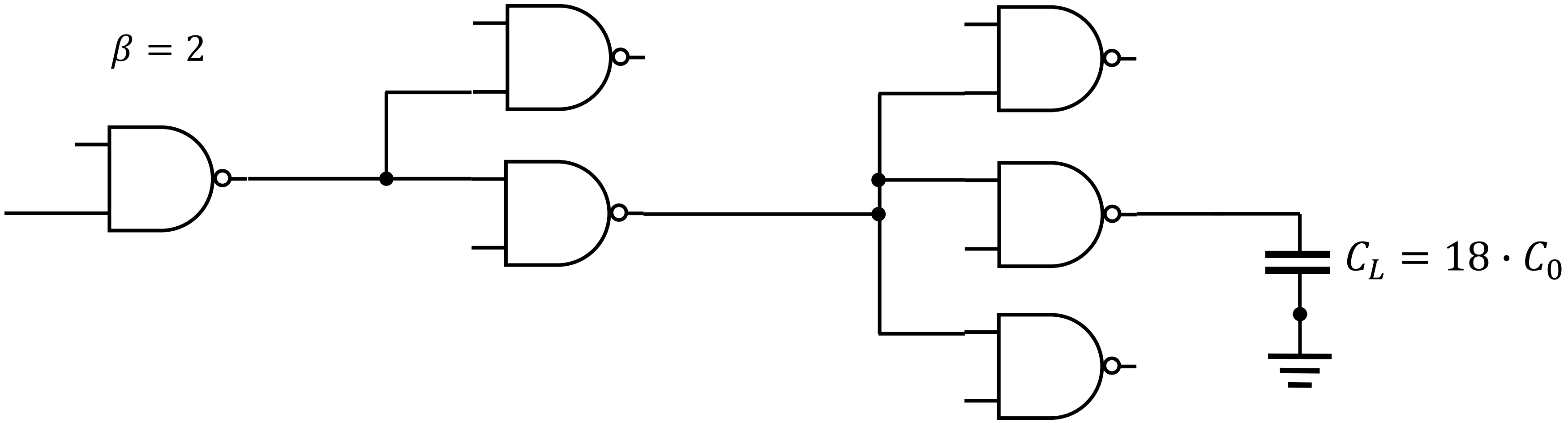
  - **Updated sizing rules include b_i** similar to g_i
    - Left-to-right: $\gamma_n = \hat{f}^{n-1} \cdot \gamma_1 \cdot \prod_{i=2}^{n} b_{i-1}^{-1} g_i^{-1} = \hat{f} \cdot \frac{\gamma_{n-1}}{b_{n-1}g_n}$
    - Right-to-left: $\gamma_n = \frac{1}{\hat{f}^{N-n+1}} \cdot \frac{C_L}{C_0^n} \cdot \prod_{i=n}^{N} b_i g_i = \frac{b_n g_{n+1}}{\hat{f}} \cdot \gamma_{n+1}$ $\quad \gamma_N = \frac{1}{\hat{f}} \cdot \frac{C_L}{C_0^N} \cdot b_N g_N$

      usually 1 unless C_L is per branch

  - Not always realistic, but many important circuits are structured and continue similarly after a branch
  - Useful also as a starting point, even when branches are finally not scaled as assumed (overdesign)

Sizing with Branches (Example)

- Consider the following circuit:



	i = 1	i = 2	i = 3
p			
g			
b			

$H =$

$G =$

$B =$

$\hat{f} = \quad =$

$D_{min} = \quad =$

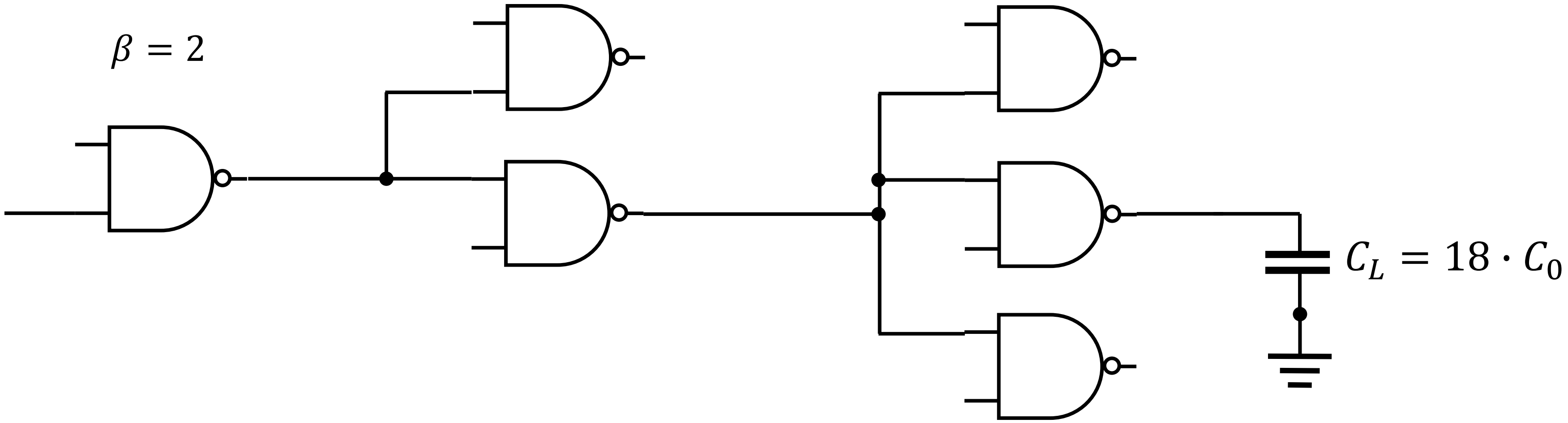
$\gamma_1 = 1$

$\gamma_2 = \quad =$

$\gamma_3 = \quad =$

Sizing with Branches (Example)

- Consider the following circuit:



		i = 1	i = 2	i = 3	
	p	2	2	2	$H = 18/4$
$N = 3$	g	4/3	4/3	4/3	$G = (4/3)^3$
	b	2	3	1	$B = 2 \cdot 3 \cdot 1$

$\hat{f} = \sqrt[3]{GBH} = 4$

$D_{min} = 6 + N \cdot \hat{f} = 22$

$\gamma_1 = 1$ $\quad \gamma_2 = \gamma_1 \frac{\hat{f}}{b_1 g_2} = \frac{3}{2}$ $\quad \gamma_3 = \gamma_2 \frac{\hat{f}}{b_2 g_3} = \frac{3}{2}$

EE-429
Fundamentals of VLSI Design

Logical Effort – Interconnect & Irregularities

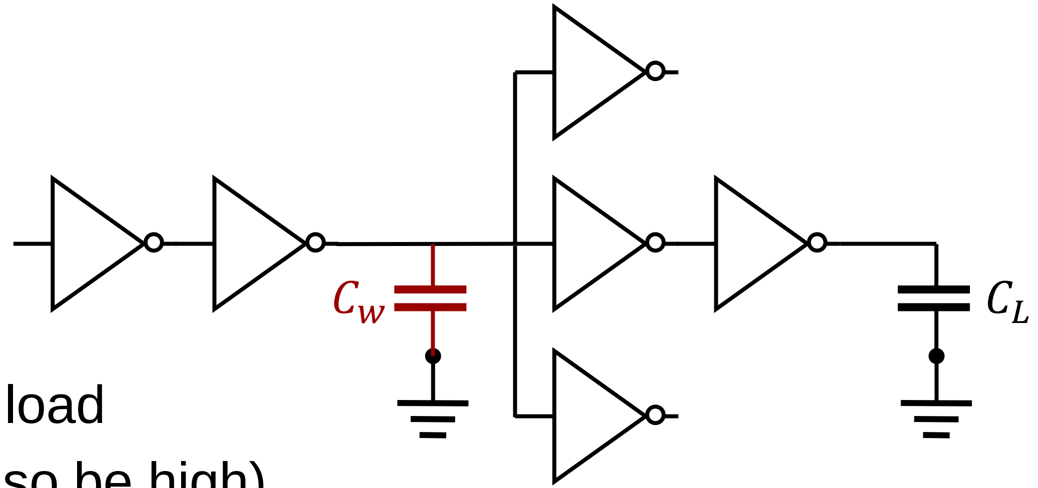
Andreas Burg

Logical Effort with Routing Capacitance (1/2)

- The logical effort model can not account for routing resistance, but routing capacitance is anyway often more relevant

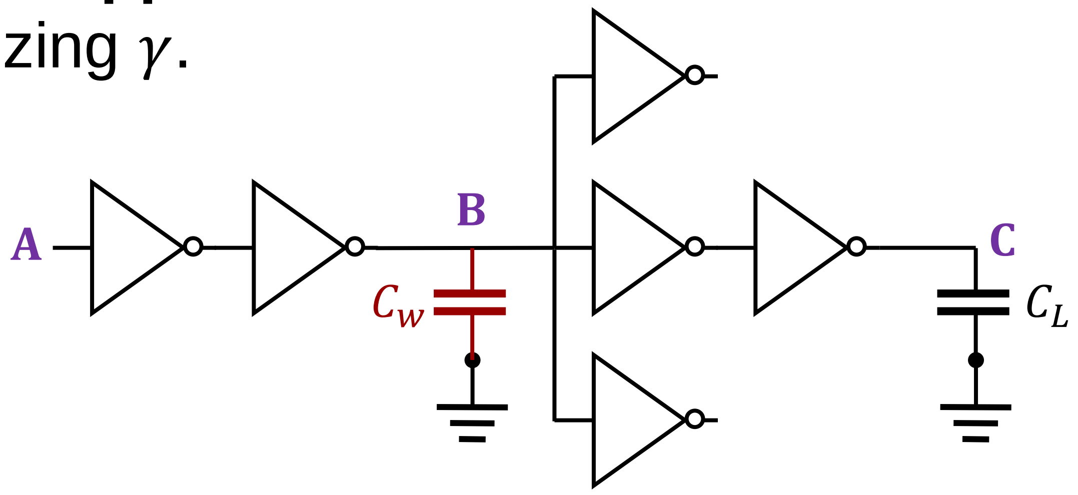
- **We often face the following scenario**
  - A **path** (to be optimized) with some random logic **contains a very long wire with a high capacitive load C_w**
  - Often, this occurs in the location of a branch with many nodes which add additional load
  - The path continues after the segment with the high load
  - The path ends with a defined load C_L (which may also be high)

  - **We would like to optimize the entire path …**



Logical Effort with Routing Capacitance (2/2)

- The **common optimization procedure is not applicable** since the wire load does not scale with the sizing γ.



- **Procedure**: multi-step iterative process

1. Ignore parasitics C_w and size all gates from **A** to **C**
    - Identify the appropriate fanout load on node **B**
    - Include any rounding-up on the drivers loading **B**

2. Optimize the partial path from **A** to **B** with the load defined by C_w and the fanout from the first optimization step.