University of Illinois at Urbana-Champaign
Dept. of Electrical and Computer Engineering

ECE 120: Introduction to Computing

Optimizing Logic Expressions

We Can Use Logical Completeness to Express Functions

Let the truth table to the right define the **function F**.

Recall that we can use the logical completeness construction to write **F** as a Boolean expression:

- This row is… **AB'C**
- And this is… **ABC'**
- And this is … **ABC**

A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

What's the Best Way to Write Function F?

So **F = AB'C + ABC' + ABC**

But we can also write **F = AB + AC**.

What about **F = A (B + C)**?

Which one is best?

A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

Your Answer Is Wrong! Choose a Metric First

The answer depends on our **choice of metric!**

How do we measure good?

Common answers for circuit design:

- area / size / cost, OR
- performance / speed, OR
- power / energy consumption, OR
- complexity / reliability.

We Use Heuristics for These Metrics

In practice, **measuring exactly is expensive** (~$50-100M for a full design, and ~$2-5M just for trying something.)

Instead, we use **heuristics**, which are ways of **estimating a metric**.

A good heuristic is
- **reasonably accurate**, and
- **monotonic** relative to a real measurement
- (so that bigger estimates mean bigger measurements).

An Area Heuristic for ECE120

Here's a **heuristic for area**:
- **Count literals** (A, A', B, B', C, C'), then
- **Add the number of operations** (not including complements for literals).

Why does it work? Remember gate structures?
- each input (literal) → two transistors
- operators into operators → two transistors

So it gives an approximate **transistor count**.

(But wires also take space!)

A Delay (Speed) Heuristic for ECE120

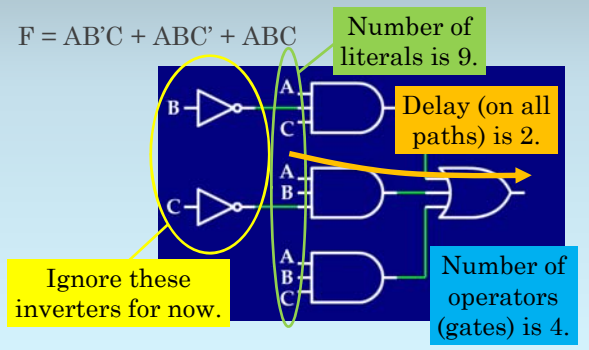
Here's a **heuristic for delay / speed**:
- **Find the maximum number of gates between any input and any output.**
- Do not include complements for literals.

Why does it work?
- Each gate takes time switch its output on/off.
- We call this time a **gate delay**.

So it gives an approximate **delay** between inputs changing and outputs changing.

A Graphical View May Make Counting Easier

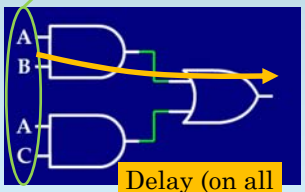
F = AB'C + ABC' + ABC



Let's Analyze the Second Form of F

F = AB + AC

Number of literals is 4.



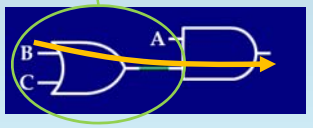
Number of operators (gates) is 3.

Delay (on all paths) is 2.

Let's Analyze the Third Form of F

F = A (B + C)

Number of literals is 3.



Number of operators (gates) is 2.

Delay (on longest paths) is 2.

The Area Heuristic Favors F = A (B + C)

Let's calculate the area heuristic for our three forms of F.

So **F = A (B + C) is the smallest design.**

Form of F	Lits	Ops	Area
AB'C + ABC' + ABC	9	4	13
AB + AC	4	3	7
A (B + C)	3	2	(5)

All Forms Are Equivalent in Delay

All designs are the same for delay.

Form of F	Lits	Ops	Area	Delay
AB'C + ABC' + ABC	9	4	13	(2)
AB + AC	4	3	7	(2)
A (B + C)	3	2	(5)	(2)

We Have a Winner: F = A (B + C)

F = A (B + C) is best by both metrics.

But the answers are not always so simple.

Sometimes no solution is best by both metrics.

- See Section 2.1.1 for a simple example.
- Later in our class, we will explore more space/time tradeoffs in design.
- In practice, tradeoffs are commonplace.
- Take a look at Section 2.1.6* for more.

What About Power and Complexity?

These two metrics are beyond our class' scope.
You'll see power in ECE385.

One heuristic for power

- uses the fact that **current flows when a transistor switches on/off**
- and uses simulation to **estimate the number of times** that happens.

Complexity is hard to measure, and is usually based on experience.

