

Karnaugh Maps

CS31

Pascal Van Hentenryck

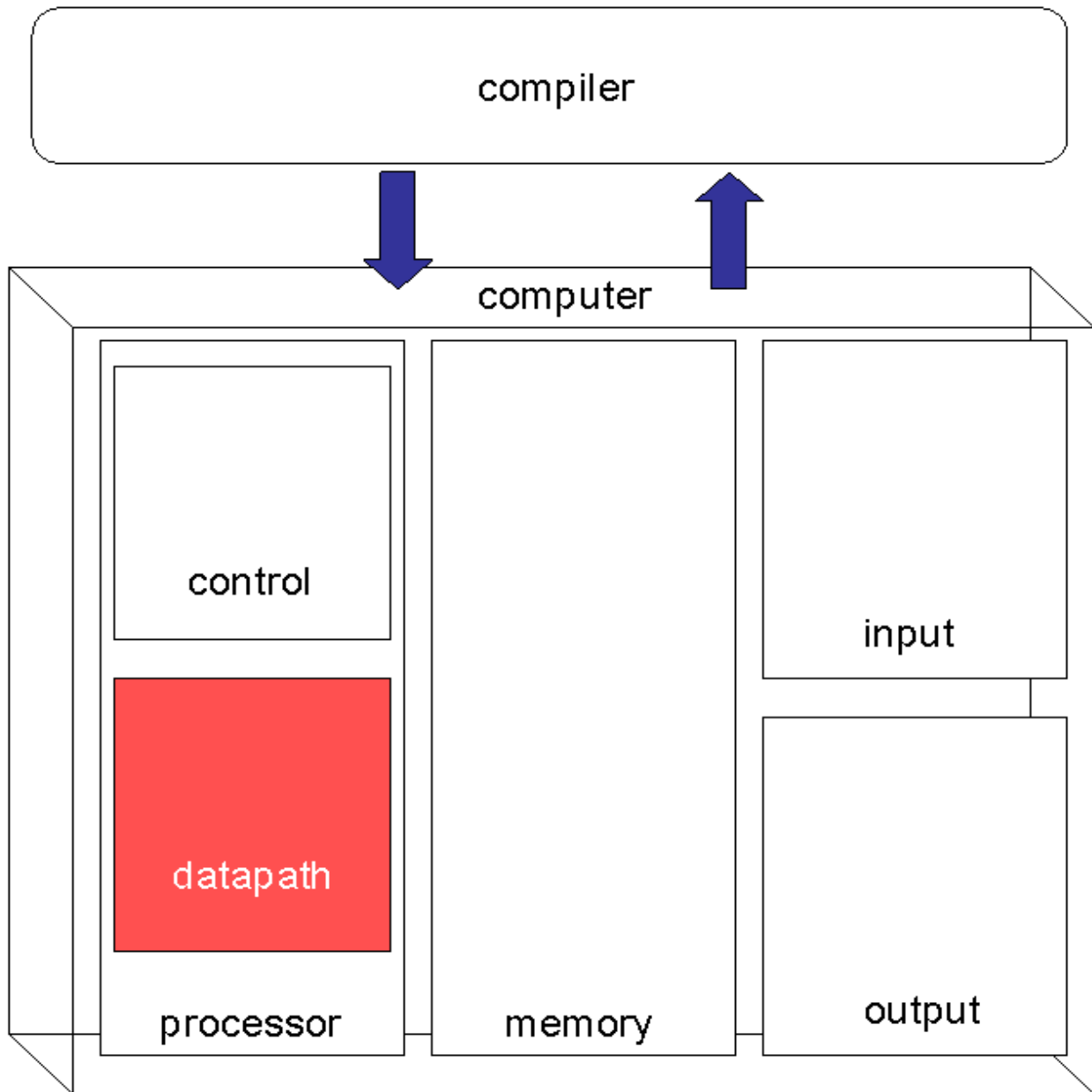


Overview

Karnaugh Maps

- Simplifying Boolean functions

The Big Picture



Abstraction Hierarchy

Programming Language

Assembly Language

Machine Language

Sequential Circuit

Combinational Circuit

Binary Value

Voltage

Minimizing Boolean Functions

The problem

- Given a truth table, find a small circuit to compute it
- The problem is very hard in general (no really efficient algorithm exists)

Possible solutions

- Put the function as a sum of products and build the corresponding circuit
- Put the function as product of sums and build the corresponding circuit

Limitation

- The size of the circuit may be far from the minimal size

Karnaugh Maps

- A manual and heuristic method to minimize the size of the circuit

Karnaugh Map

Another Presentation of a Truth Table

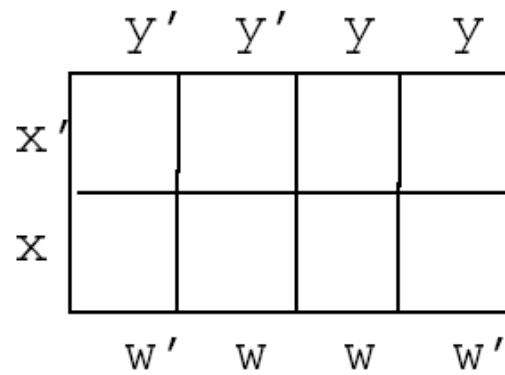
a	b	c
0	0	1
0	1	1
1	0	0
1	1	0

The Karnaugh map looks like this

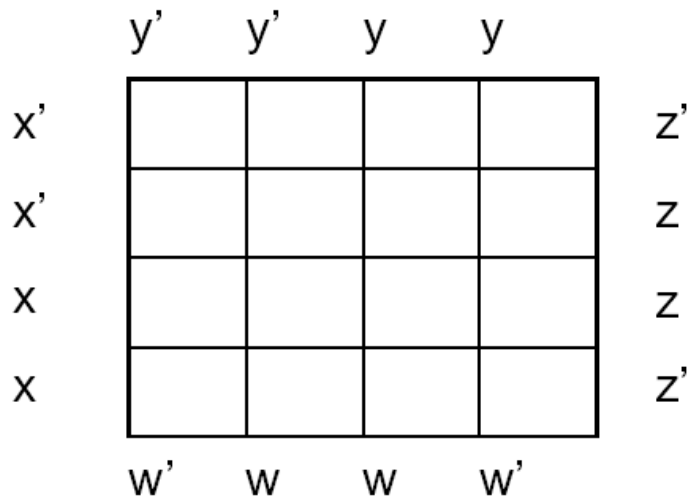
	b'	b
a'	1	1
a	0	0

Karnaugh Map

3 variables



4 variables



Karnaugh Maps

Property

- Adjacent entries differ by at most one variable

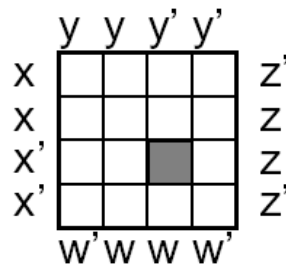
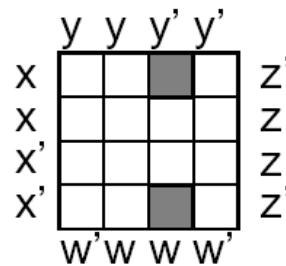
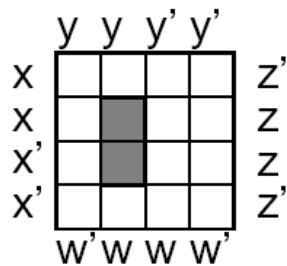
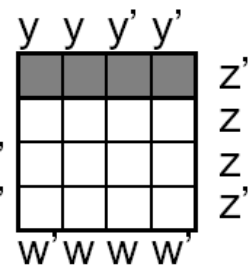
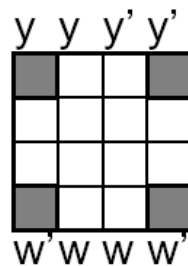
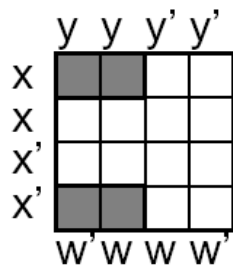
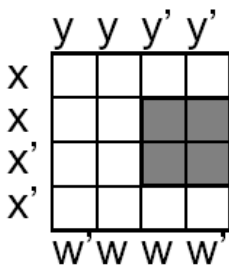
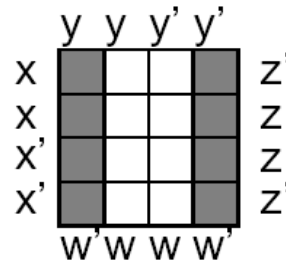
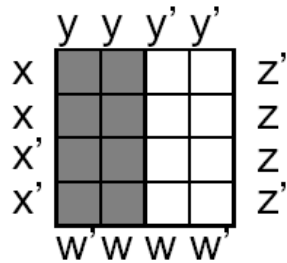
	y'	y'	y	y
x'				
x'				
x				
x				
	w'	w	w	w'

- The whole table is wrapping up on itself. The end of a row is adjacent to the beginning of the row and so on.

	y	y	y'	y'	
x					z'
x					z
x'					z
x'					z'
	w'	w	w	w'	

Boxes

Boxes of sizes 1,2,4,8 or 16



Minimization

The basic strategy

1. Draw the Karnaugh map
2. Fill it with the truth table
3. Cover all the 1s with boxes of size 1, 2, 4, 8, 16. ... (an entry can be covered by several boxes if needed)
4. Generate a product for each box; each element of the product corresponds to a variable which stays constant over the box. It is the variable if the variable stays at 1 and the negation of the variable otherwise.
5. The result is the sum of all these products

The main goal

- As few boxes as possible
- The biggest possible boxes

A simplification

Another Presentation of a Truth Table

a	b	c
0	0	1
0	1	1
1	0	0
1	1	0

The Karnaugh map looks like this

	b'	b
a'	1	1
a	0	0

What is the boolean function?

A simplification

Another Presentation of a Truth Table

a	b	c
0	0	1
0	1	1
1	0	0
1	1	0

The Karnaugh map looks like this

	b'	b
a'	1	1
a	0	0

What is the boolean function?

$$f(a,b)=a'$$

Another Simplification Problem

x	y	z	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

Sum of Products gives us:

Another Simplification Problem

x	y	z	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

Sum of Products gives us:

$$f(x,y,z) = x'yz + xy'z' + xyz' + xyz$$

Karnaugh map:

	y'	y'	y	y
x'	0	0	1	0
x	1	0	1	1
	z'	z	z	z'

Another Simplification Problem

x	y	z	f
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

Sum of Products gives us:

$$f(x,y,z) = x'yz + xy'z' + xyz' + xyz$$

Karnaugh map:

	y'	y'	y	y
x'	0	0	1	0
x	1	0	1	1
	z'	z	z	z'

$$f(x,y,z) = yz + xz'$$

A Really Big One

	w'	w'	w	w	
x'	1	1	0	1	y'
x'	1	1	1	1	y
x	0	1	1	1	y
x	0	0	0	1	y'
	z'	z	z	z'	

Don't Cares

Sometimes, it doesn't matter exactly which output is generated in a certain situation.

A don't care value is represented by a x and we can choose any value that is convenient to us.

	y'	y'	y	y
x'	1	0	x	x
x	x	x	0	1
	z'	z	z	z'

What do we choose?

By setting the left and right x 's to 1 we can make a 2x2 square, reducing the function to: $f(x,y,z)=z'$

Don't Cares

Sometimes, it doesn't matter exactly which output is generated in a certain situation.

A don't care value is represented by a x and we can choose any value that is convenient to us.

	y'	y'	y	y
x'	1	0	x	1
x	1	x	0	1
	z'	z	z	z'

What do we choose?

By setting the left and right x 's to 1 we can make a 2x2 square, reducing the function to: $f(x,y,z)=z'$

We Still Don't Care!

	w'	w'	w	w	
x'	0	0	0	0	y'
x'	1	1	0	0	y
x	0	1	1	0	y
x	0	1	0	0	y'
	z'	z	z	z'	

What if the x's were 0's?

$$f(w,x,y,z)=(w'x'y) + (w'xz) + (xyz)$$

We Still Don't Care!

	w'	w'	w	w	
x'	0	0	0	0	y'
x'	1	1			y
x	0	1	1		y
x	0	1			y'
	z'	z	z	z'	

What if the x's were 0's?

$$f(w,x,y,z)=(w'x'y) + (w'xz) + (xyz)$$

What if they're don't cares?

We Still Don't Care!

	w'	w'	w	w	
x'	0	0	0	0	y'
x'	1	1			y
x	0	1	1		y
x	0	1			y'
	z'	z	z	z'	

What if the x's were 0's?

$$f(w,x,y,z)=(w'x'y) + (w'xz) + (xyz)$$

What if they're don't cares?

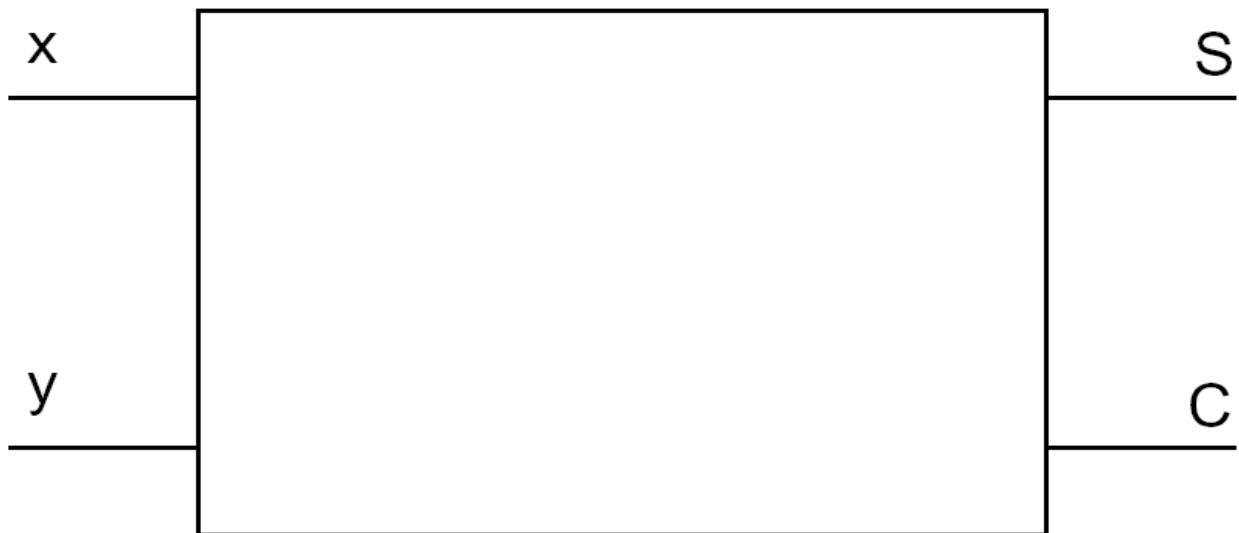
$$f(w,x,y,z)=(x'y) + (xz)$$

How many operations do we save?

Including negations, $11-4=7$

Half Adder

Given two inputs, generate the sum and carry.



x	y	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Full Adder

Add three inputs, generating the sum and carry.



x	y	z	S	C
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Simplification for FA

Karnaugh map for S

	y	y	y'	y'
x	0	1	0	1
x'	1	0	1	0
	z'	z	z	z'

Karnaugh map for C

	y	y	y'	y'
x	1	1	1	0
x'	0	1	0	0
	z'	z	z	z'

Chaining Adders

We can add two 4-bit numbers
by chaining full adders.

