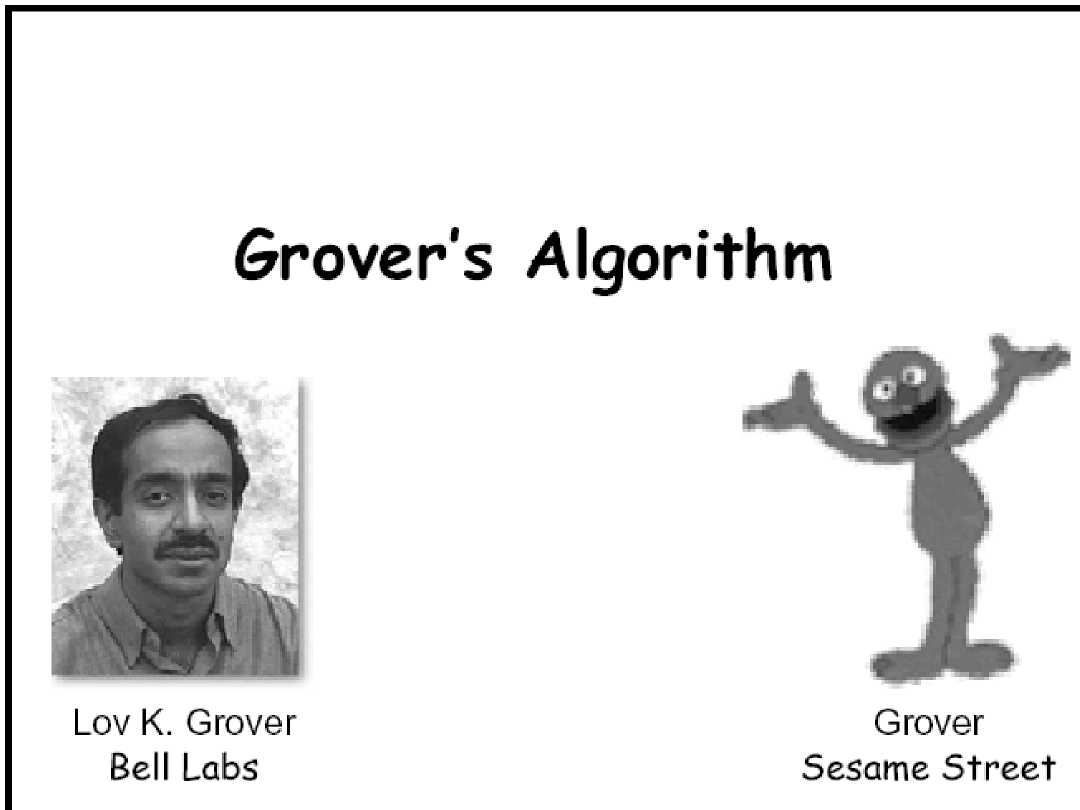
Grover’s Algorithm in Machine Learning and Optimization Applications

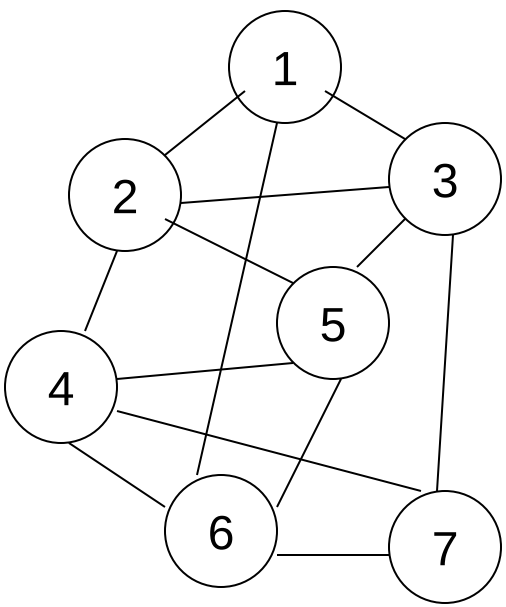
Grover Algorithm Reminder in new light



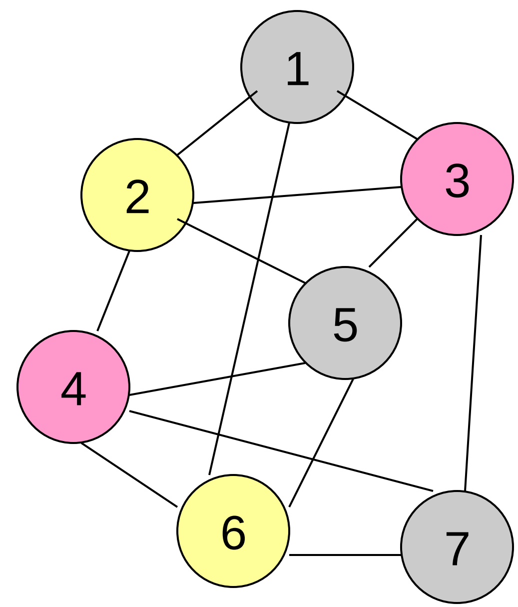
Graph Coloring

- Building oracle for graph coloring is a better explanation of Grover than database search.
- This is not an optimal way to do graph coloring but explains well the principle of building oracles.

The Graph Coloring Problem

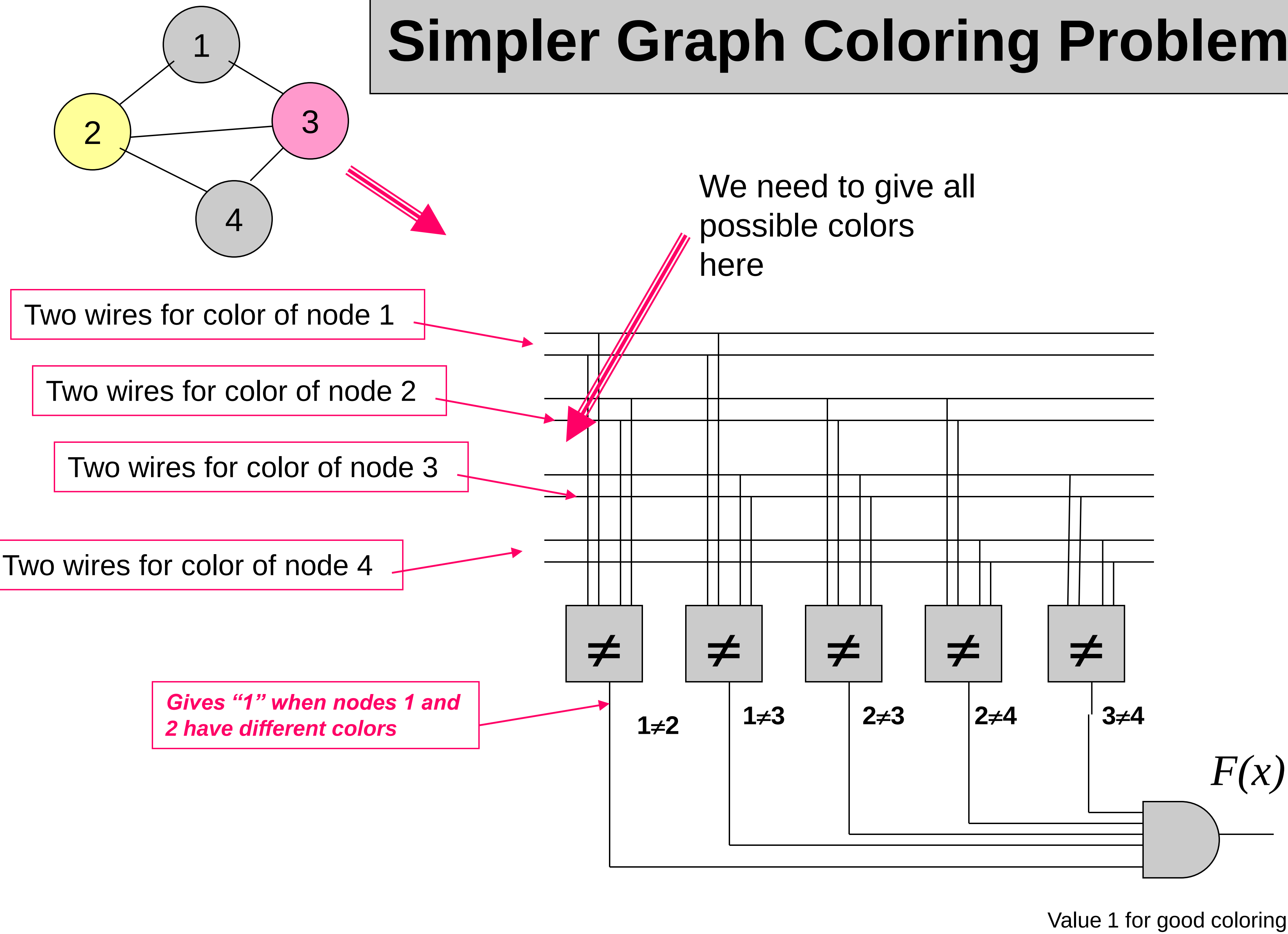


Color every node with a color. Every two nodes that share an edge should have different colors. Number of colors should be minimum



This graph is 3-colorable

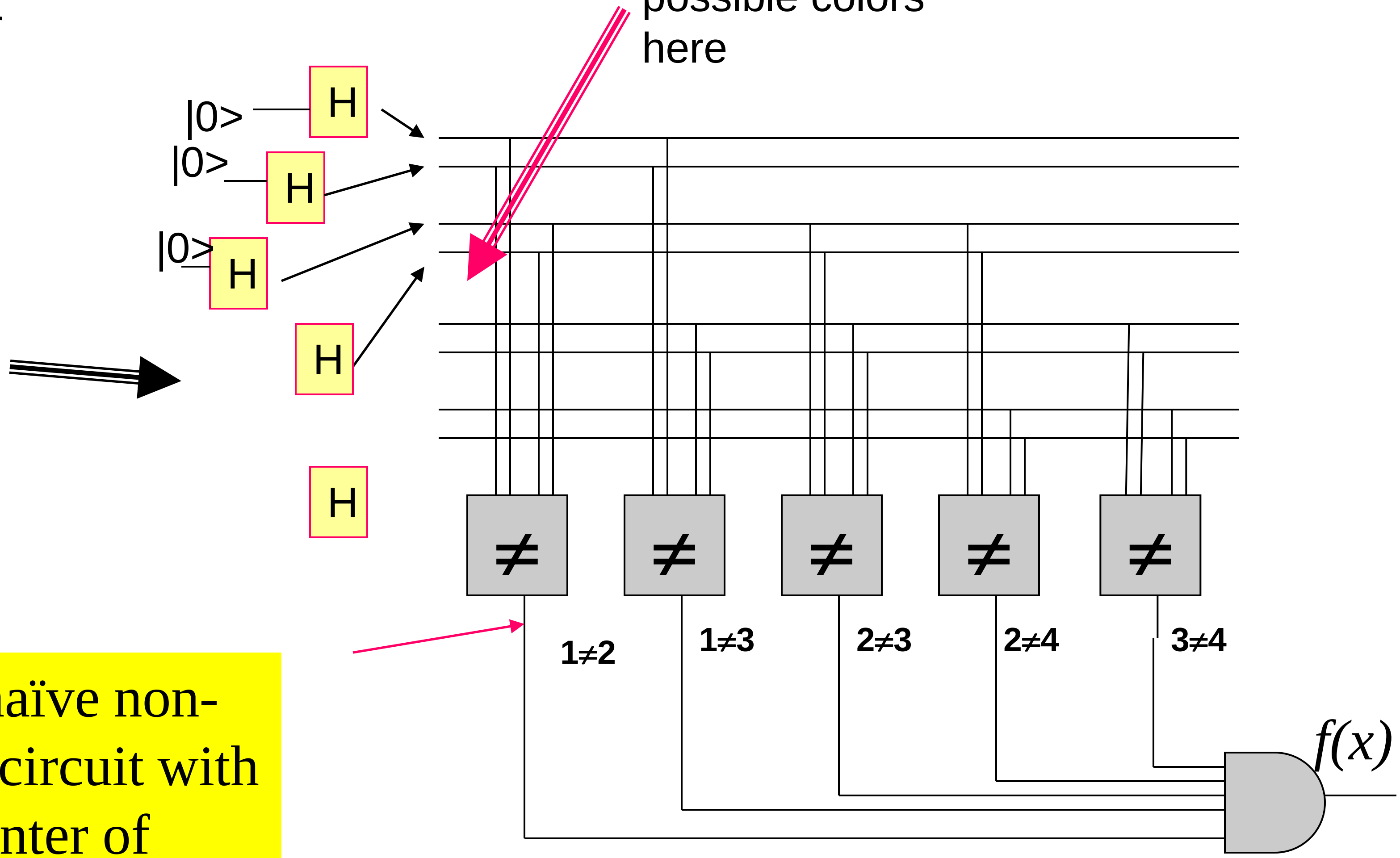
Simpler Graph Coloring Problem



Simpler Graph Coloring Problem

Give Hadamard for each wire to get superposition of all state, which means the set of all colorings

We need to give all possible colors here



Value 1 for good coloring

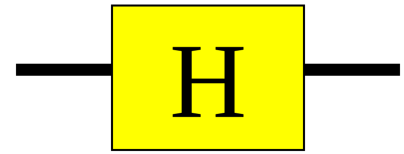
Discuss naïve non-quantum circuit with a full counter of minterms

Now we will generate whole Kmap at once using quantum properties - Hadamard

Hadamard Transform

$$H = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix}$$

Single qubit

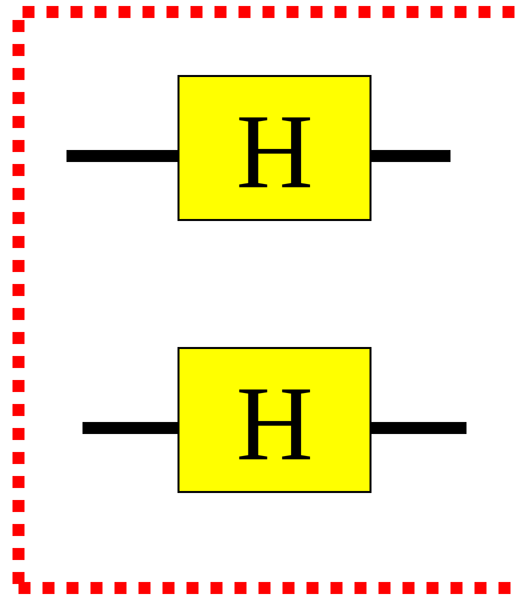


$$\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \otimes \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} =$$

$= 1/2$

1	1	1	1
1	-1	1	-1
1	1	-1	-1
1	-1	-1	1

Here I calculated **Kronecker product** of two Hadamards



Parallel connection of two Hadamard gates is calculated by Kronecker Product (tensor product)

- As we remember, these are transformations of Hadamard gate:

Motivating calculations for 3 variables

$|0\rangle$ — $\boxed{H}$ — $|0\rangle + |1\rangle$ $\qquad$ $|1\rangle$ — $\boxed{H}$ — $|0\rangle - |1\rangle$

$$\boxed{\begin{array}{l}\text{In general:}\\ |x\rangle \;—\; \boxed{H} \;—\; |0\rangle + (-1)^{x}\, |1\rangle\end{array}}$$

For 3 bits, vector of 3 Hadamards works as follows:

$$|abc\rangle \rightarrow \quad (|0\rangle+(-1)^a|1\rangle) \;\; (|0\rangle+(-1)^b|1\rangle)\,(|0\rangle+(-1)^c|1\rangle) =$$

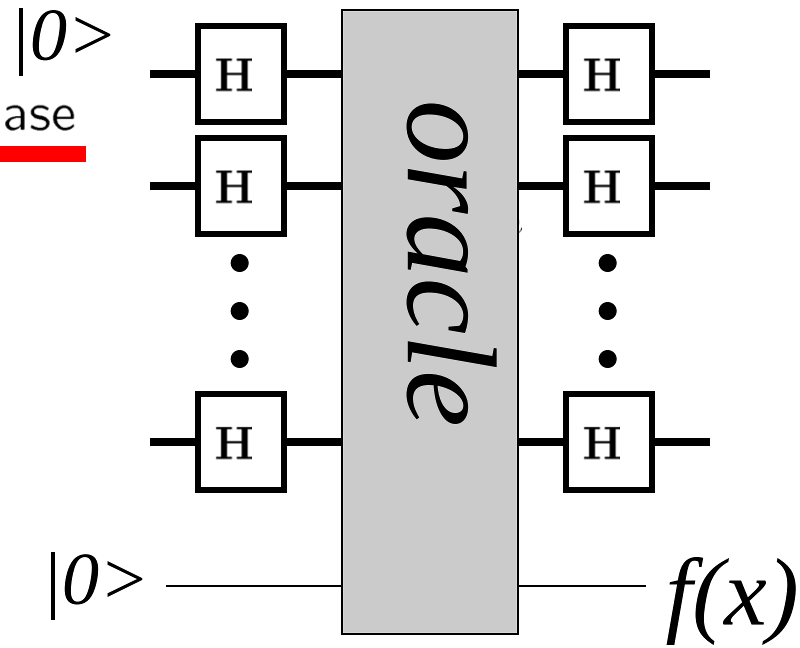
From multiplication

$$|000\rangle +(-1)^c\, |001\rangle +(-1)^b\, |001\rangle+(-1)^{b+c}\, |001\rangle 000\rangle +(-1)^a\, |001\rangle +$$

$$(-1)^{a+c}\, |001\rangle + (-1)^{a+b}\, |001\rangle\; (-1)^{a+b+c}\, |001\rangle$$

Information about $f(x)$ is in the phase

$$\frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} (-1)^{f(x)} |x\rangle |1\rangle$$

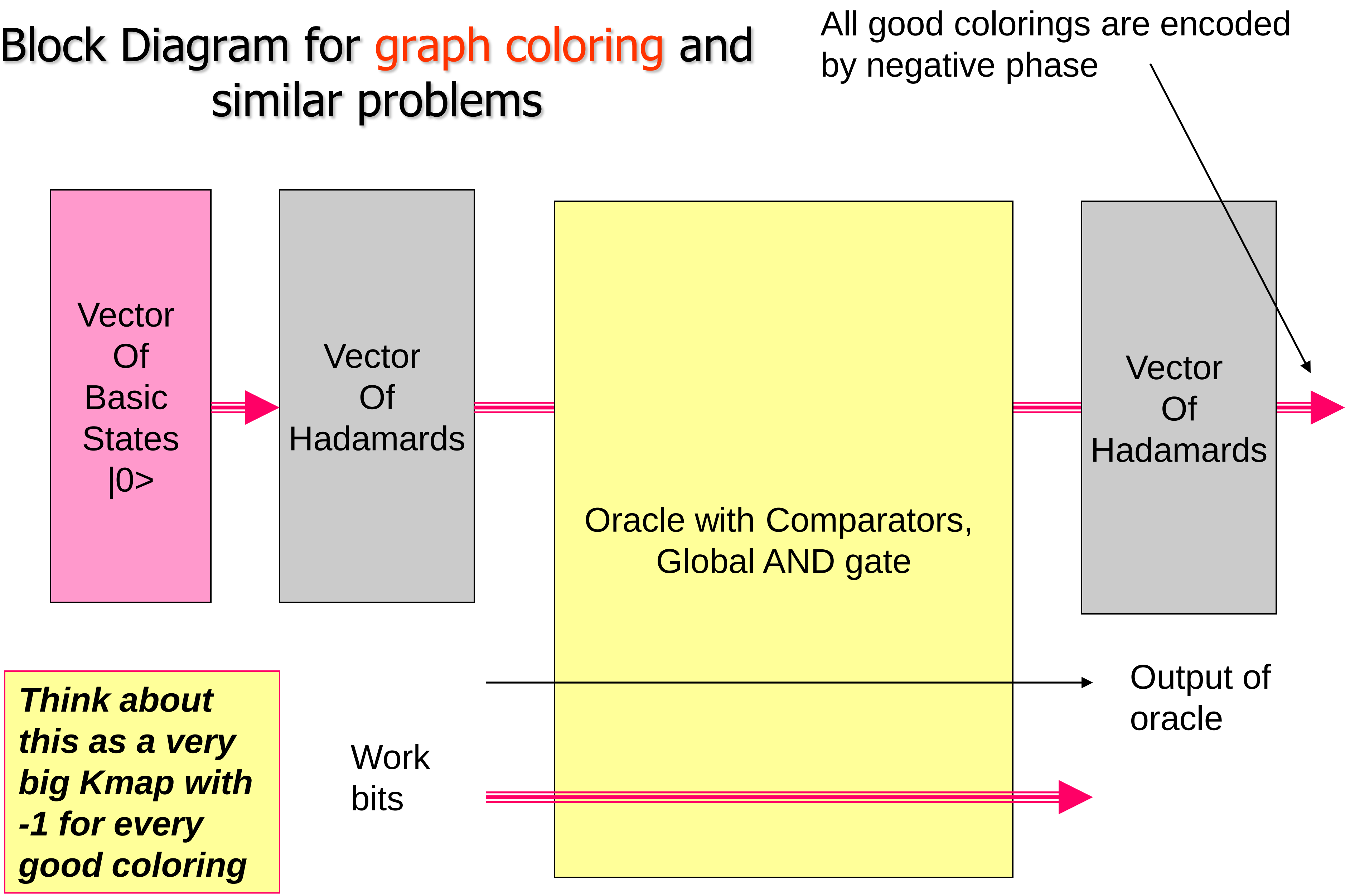


This is like a Kmap with every true minterm (1) encoded by -1

And every false minterm (0) encoded by 1

<u>We can say that Hadamard gates before the oracle create the Kmap of the function, setting the function in each of its possible minterms (cells) in parallel</u>

Block Diagram for graph coloring and similar problems

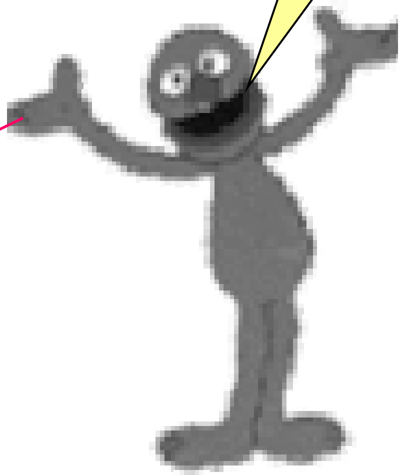


What Grover algorithm does?

- Grover algorithm looks to a very big Kmap and tells where is the -1 in it.

1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	-1	1	1	1	1	1
1	1	1	1	1	1	1	1

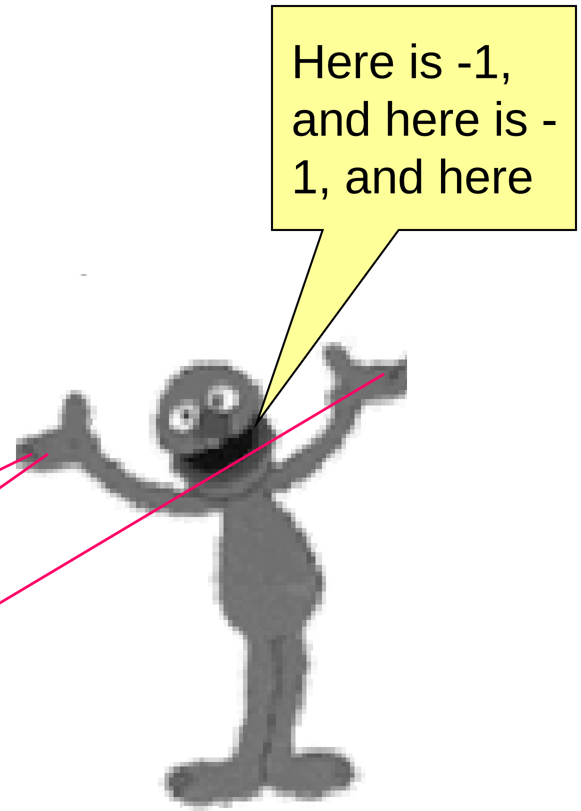
Here is -1



What "*Grover for multiple solutions*" algorithm does?

- Grover algorithm looks to a very big Kmap and tells where is the -1 in it.
- "Grover for many solutions" will tell all solutions.

1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	-1	1	1	1	1	1
1	1	1	-1	-1	1	1	1



Variants of Grover

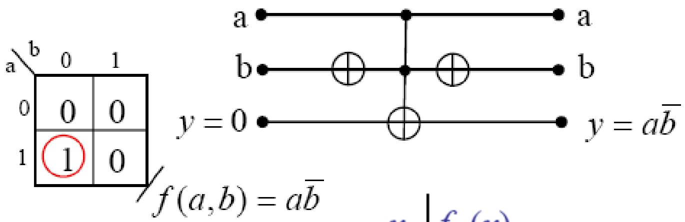
- With this oracle the "Grover algorithm for many solutions" will find all good colorings of the graph.
- If we want to find the coloring, that is good and in addition has less than K colors, we need to add the cost comparison circuit to the oracle.
- Then the oracle's answers will be "one" only if the coloring is good and has less colors than K.
- The oracle thus becomes more complicated but the Grover algorithm can be still used.

A practical Example

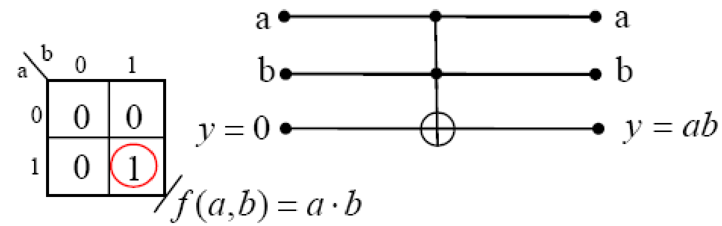
- This presentation shows clearly how to perform a so called **1 in 4 search**
- We start out with the basics

1 in 4 search

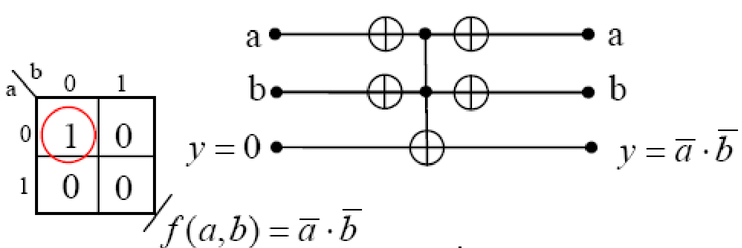
Pick your needle and I will find you a haystack



x	$f_{10}(x)$
00	0
01	0
10	1
11	0

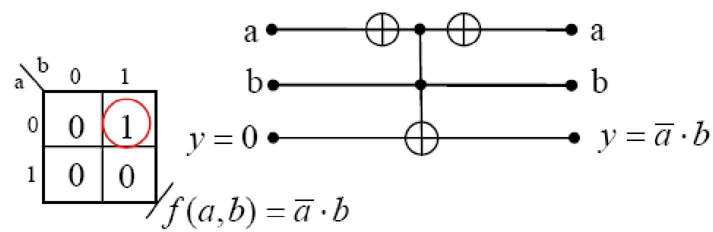


x	$f_{11}(x)$
00	0
01	0
10	0
11	1



x	$f_{00}(x)$
00	1
01	0
10	0
11	0

The point of this slide is to show examples of 4 different oracles. Grovers search can tell between these oracles in a single iteration, classically we would need 3 iterations.



x	$f_{01}(x)$
00	0
01	1
10	0
11	0

Properties of the oracle

Let $f: \{0,1\}^2 \rightarrow \{0,1\}$ have the property that there is exactly one $x \in \{0,1\}^2$ for which $f(x) = 1$

Goal: find $x \in \{0,1\}^2$ for which $f(x) = 1$

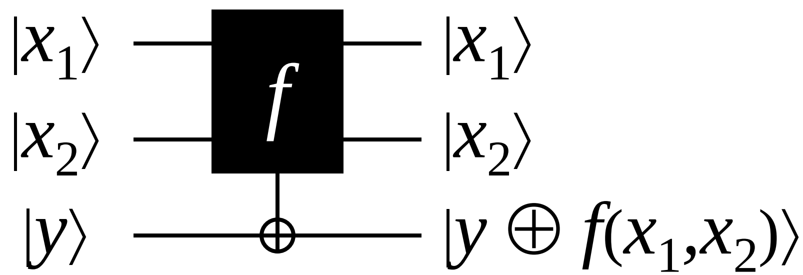
Classically: 3 queries are necessary

Quantumly: ?

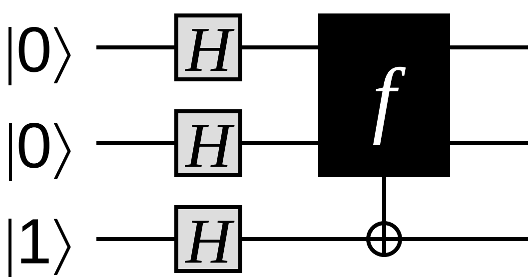
Only after 3 tests can we determine with certainty that the oracles is 1 for only a single input value x

A 1-4 search can chose between 4 oracles **in one iteration**

Black box for 1-4 search:



Start by creating phases in superposition of all inputs to f:



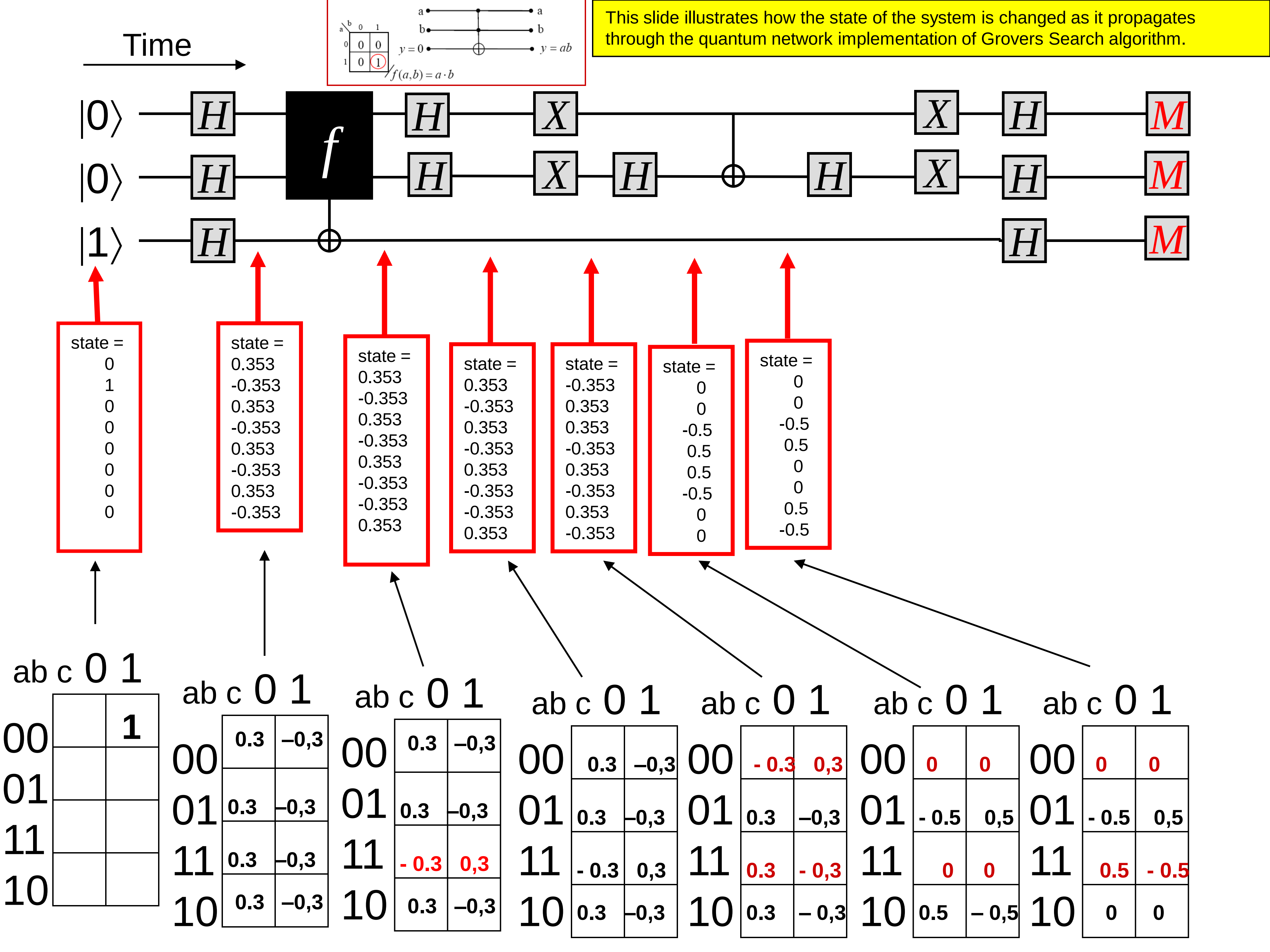
Input state to query:

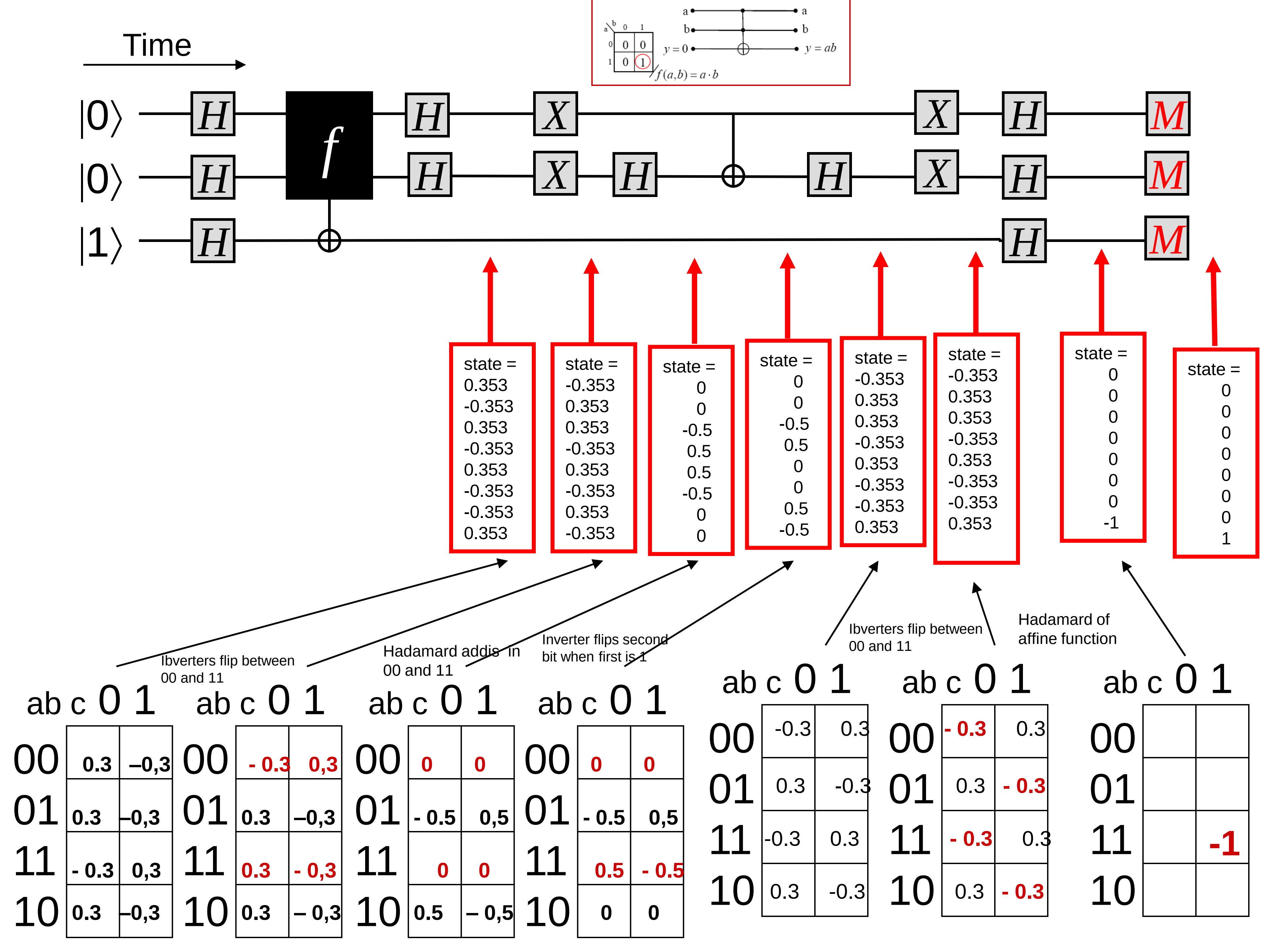
$$(|00\rangle + |01\rangle + |10\rangle + |11\rangle)(|0\rangle - |1\rangle)$$

Output state:

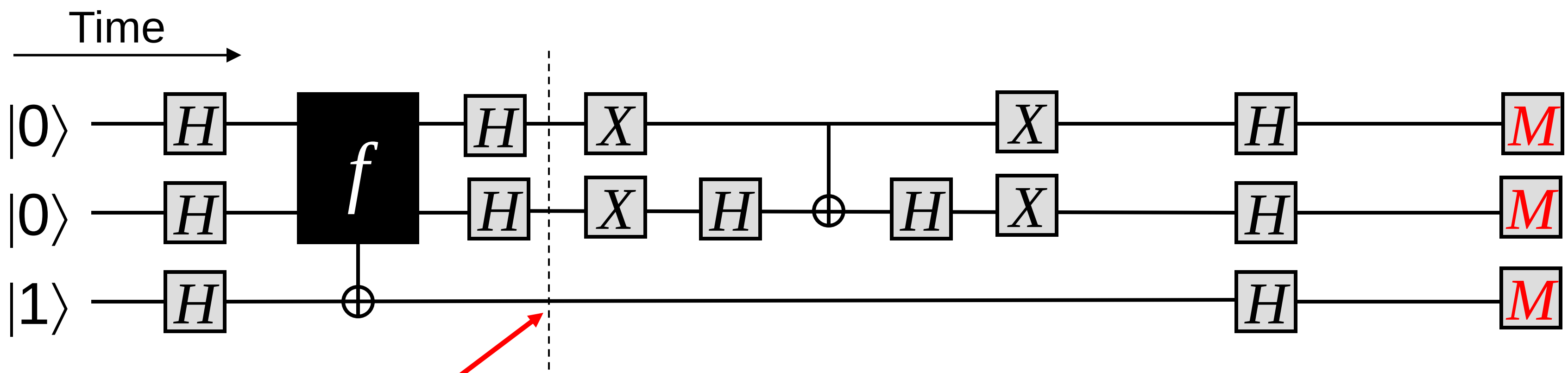
$$((-1)^{f(00)}|00\rangle + (-1)^{f(01)}|01\rangle + (-1)^{f(10)}|10\rangle + (-1)^{f(11)}|11\rangle)(|0\rangle - |1\rangle)$$

Here we clearly see the Kmap encoded in phase – the main property of many quantum algorithms


This slide illustrates how the state of the system is changed as it propagates through the quantum network implementation of Grovers Search algorithm.
Time
a b 0 1
0 0 0
1 0 1
f(a,b) = a·b
a a
b b
y = 0
y = ab
|0⟩
|0⟩
|1⟩
H
f
X
M
state = 0 1 0 0 0 0 0 0
state = 0.353 -0.353 0.353 -0.353 0.353 -0.353 0.353 -0.353
state = 0.353 -0.353 0.353 -0.353 0.353 -0.353 -0.353 0.353
state = 0.353 -0.353 0.353 -0.353 0.353 -0.353 -0.353 0.353
state = -0.353 0.353 0.353 -0.353 0.353 -0.353 0.353 -0.353
state = 0 0 -0.5 0.5 0.5 -0.5 0 0
state = 0 0 -0.5 0.5 0 0 0.5 -0.5
ab c 0 1
00 | | 1
01 | |
11 | |
10 | |
ab c 0 1
00 | 0.3 | –0,3
01 | 0.3 | –0,3
11 | 0.3 | –0,3
10 | 0.3 | –0,3
ab c 0 1
00 | 0.3 | –0,3
01 | 0.3 | –0,3
11 | - 0.3 | 0,3
10 | 0.3 | –0,3
ab c 0 1
00 | 0.3 | –0,3
01 | 0.3 | –0,3
11 | - 0.3 | 0,3
10 | 0.3 | –0,3
ab c 0 1
00 | - 0.3 | 0,3
01 | 0.3 | –0,3
11 | 0.3 | - 0,3
10 | 0.3 | – 0,3
ab c 0 1
00 | 0 | 0
01 | - 0.5 | 0,5
11 | 0 | 0
10 | 0.5 | – 0,5
ab c 0 1
00 | 0 | 0
01 | - 0.5 | 0,5
11 | 0.5 | - 0.5
10 | 0 | 0



Time
a b 0 1
0 0 0
1 0 1
a
b
y = 0
y = ab
f(a,b) = a · b
|0⟩
|0⟩
|1⟩
H
f
X
M
state = 0.353 -0.353 0.353 -0.353 0.353 -0.353 -0.353 0.353
state = -0.353 0.353 0.353 -0.353 0.353 -0.353 0.353 -0.353
state = 0 0 -0.5 0.5 0.5 -0.5 0 0
state = 0 0 -0.5 0.5 0 0 0.5 -0.5
state = -0.353 0.353 0.353 -0.353 0.353 -0.353 -0.353 0.353
state = -0.353 0.353 0.353 -0.353 0.353 -0.353 -0.353 0.353
state = 0 0 0 0 0 0 0 -1
state = 0 0 0 0 0 0 0 1
Ibverters flip between 00 and 11
Hadamard addis in 00 and 11
Inverter flips second bit when first is 1
Ibverters flip between 00 and 11
Hadamard of affine function
ab c 0 1
00 0.3 –0,3
01 0.3 –0,3
11 - 0.3 0,3
10 0.3 –0,3
ab c 0 1
00 - 0.3 0,3
01 0.3 –0,3
11 0.3 - 0,3
10 0.3 – 0,3
ab c 0 1
00 0 0
01 - 0.5 0,5
11 0 0
10 0.5 – 0,5
ab c 0 1
00 0 0
01 - 0.5 0,5
11 0.5 - 0.5
10 0 0
ab c 0 1
00 -0.3 0.3
01 0.3 -0.3
11 -0.3 0.3
10 0.3 -0.3
ab c 0 1
00 - 0.3 0.3
01 0.3 - 0.3
11 - 0.3 0.3
10 0.3 - 0.3
ab c 0 1
00
01
11 -1
10



$$|\psi_{00}\rangle = -\,|00\rangle + |01\rangle + |10\rangle + |11\rangle$$

$$|\psi_{01}\rangle = +\,|00\rangle - |01\rangle + |10\rangle + |11\rangle$$

$$|\psi_{10}\rangle = +\,|00\rangle + |01\rangle - |10\rangle + |11\rangle$$

$$|\psi_{11}\rangle = +\,|00\rangle + |01\rangle + |10\rangle - |11\rangle$$

After Hadamard the solution is "known" in Hilbert space by having value -1. But it is hidded from us

The state corresponding to the input to the oracle that has a output result of 1 is 'tagged' with a negative 1.

This was a special case where we could transform the state vector without repeating the oracle.

In general we have to repeat the oracle – general Grover

Reed-Muller Transform Reminder

- Definition: for a function f, the Reed-Muller transform pair is given by :

$$s = R(n) \times f \quad \text{and} \quad f = R^{-1}(n) \times s$$

$$\text{where } R(n) = \otimes_i R(1), \; i = 1, 2, \ldots, n$$

$$R^{-1}(n) = \otimes_i R^{-1}(1), \; i = 1, 2, \ldots, n$$

$$R(1) = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$$

- The R-M matrix for two variables is

$$R(2) = \otimes_i R(1) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

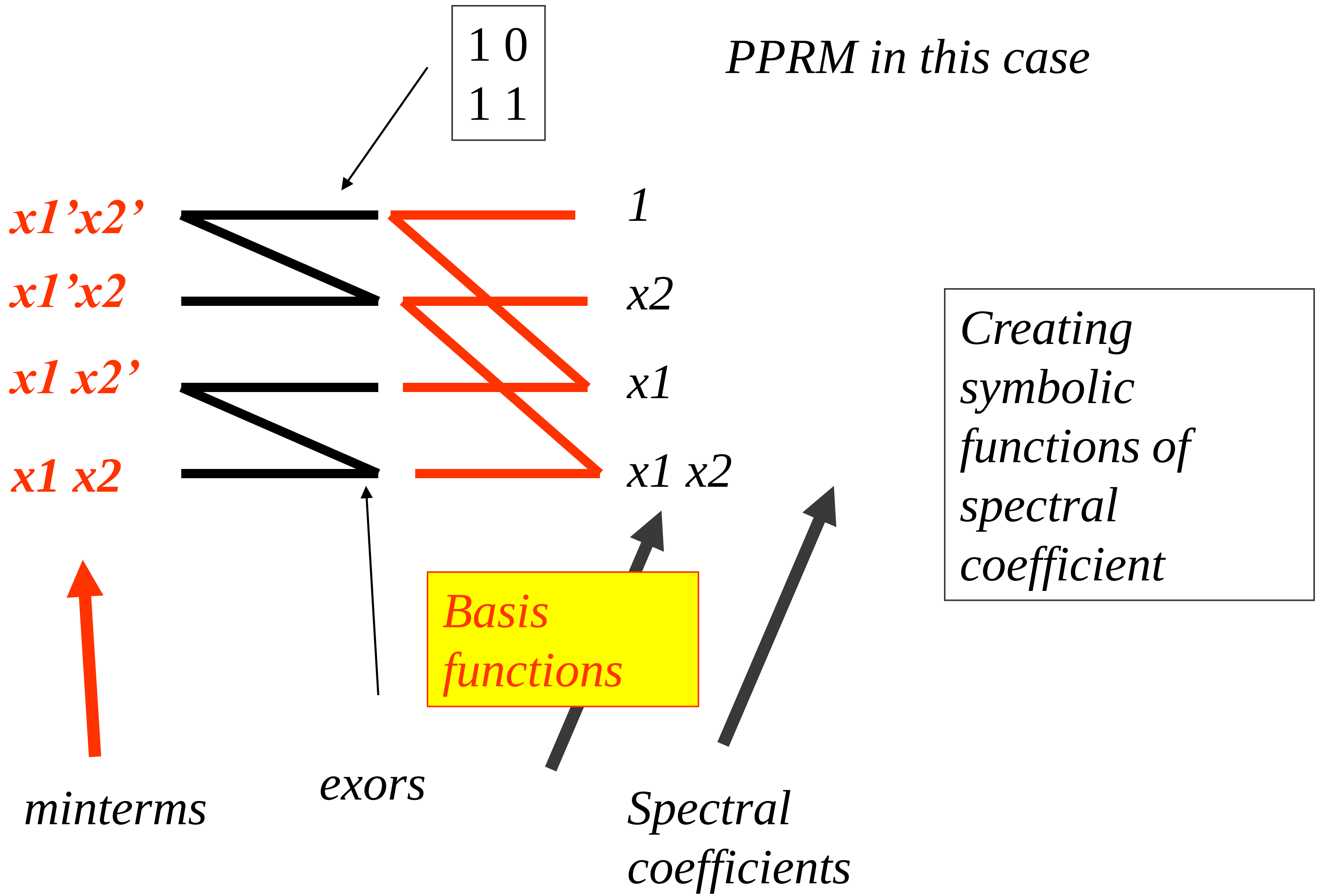
FPRM

- Functions can be represented as a Reed-Muller expansion of a given polarity using a collection of conjunctive terms joined by the modulo-additive operator such as

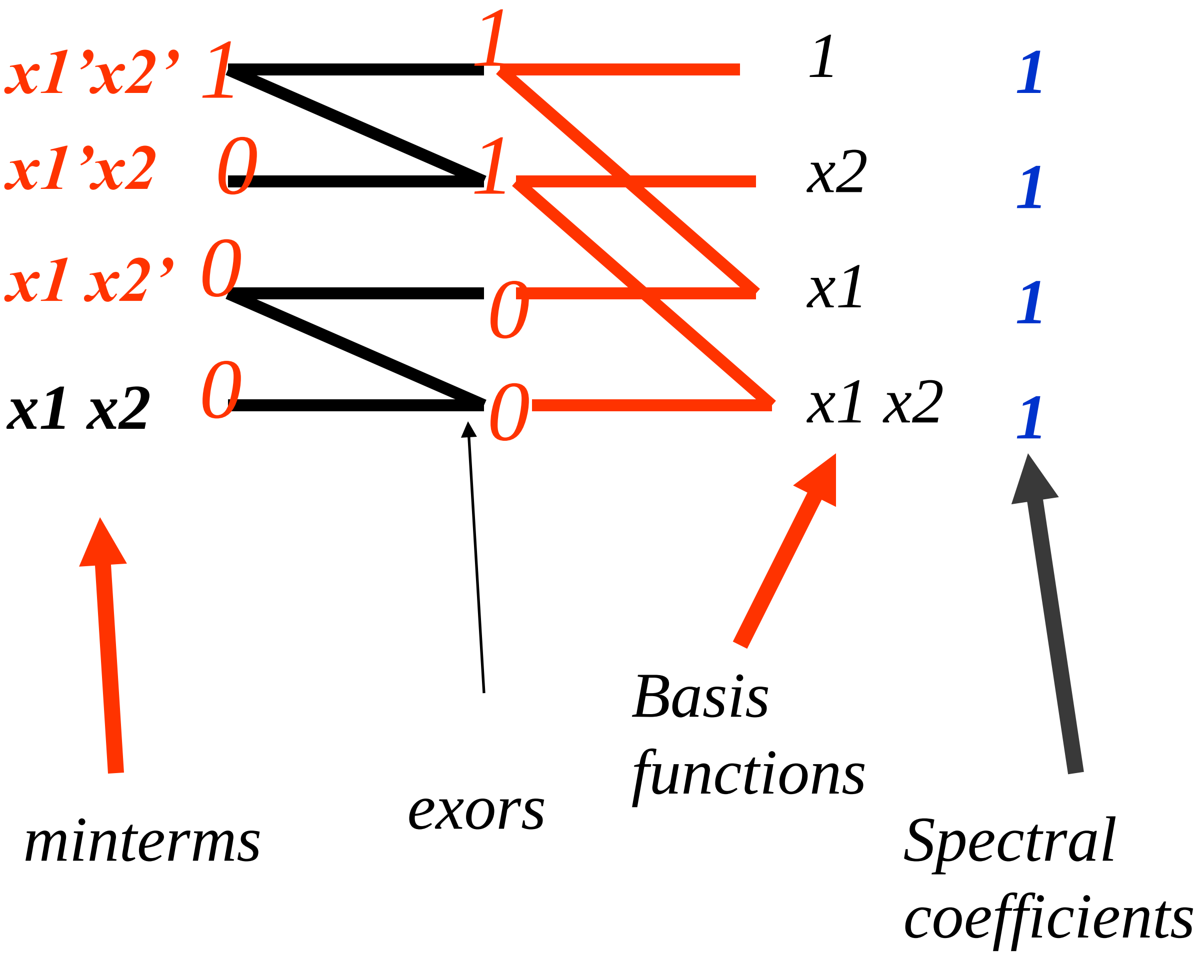
$$F = a_0 1 \oplus a_1 \dot{x}_1 \oplus a_2 \dot{x}_2 \oplus a_3 \dot{x}_3 \oplus a_{12} \dot{x}_1 \dot{x}_2$$
$$\oplus a_{13} \dot{x}_1 \dot{x}_3 \oplus a_{23} \dot{x}_2 \dot{x}_3 \oplus a_{123} \dot{x}_1 \dot{x}_2 \dot{x}_3$$

where $a_i \in \{0,1\}$

How to use this? **FPRM butterfly**



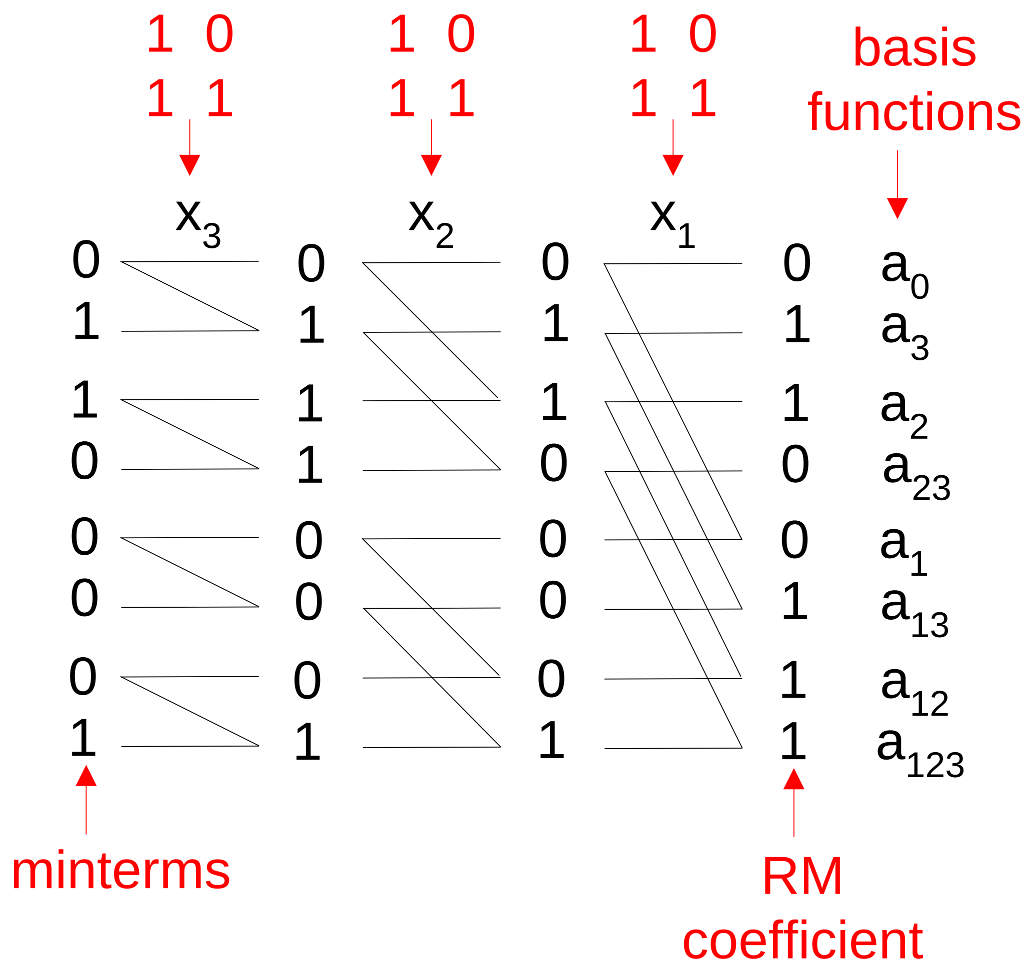
How to use this? **FPRM butterfly**



Calculating numerical values of spectral coefficients from values of function vector (minterms)

FPRM Butterfly

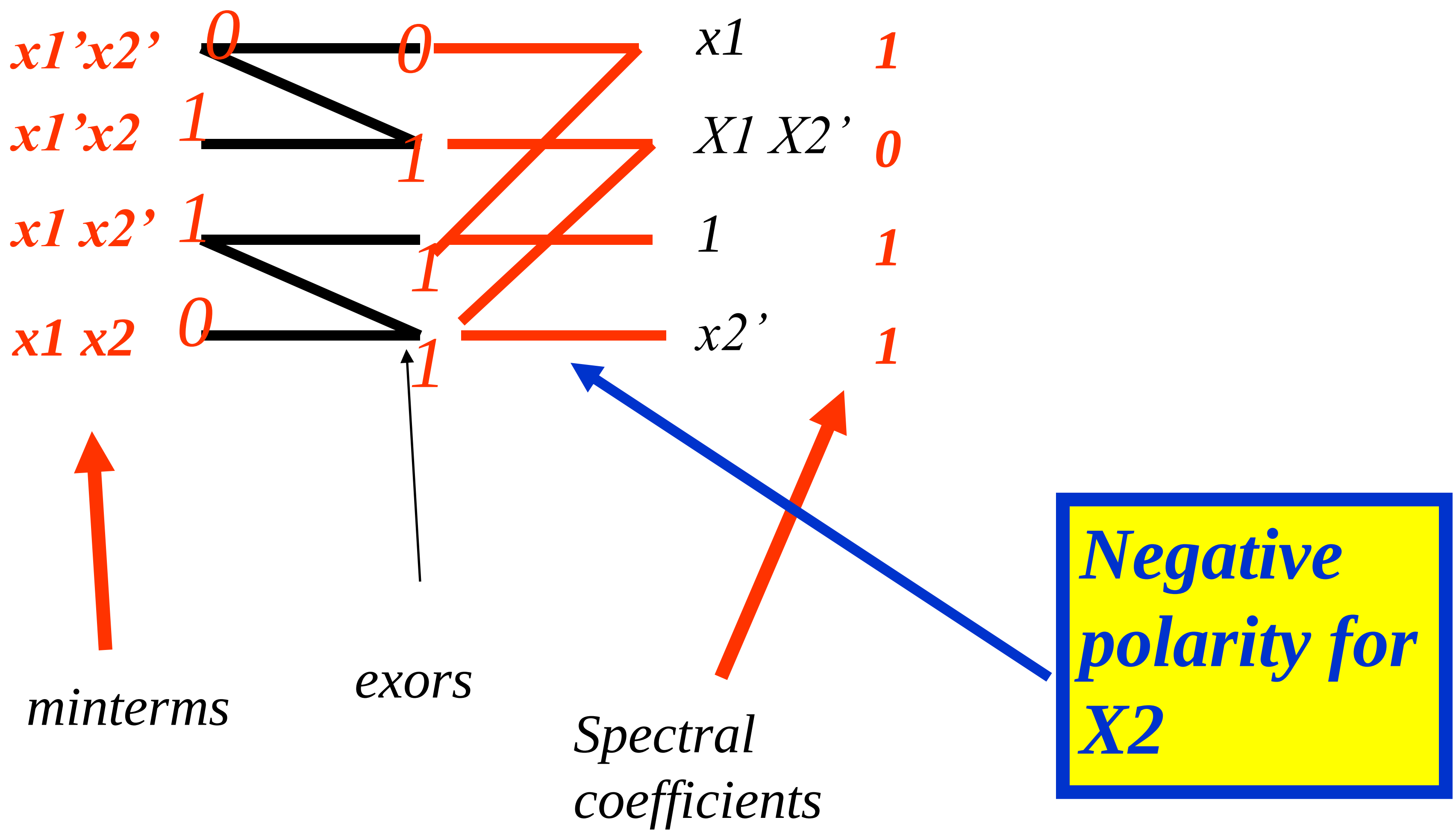
❖ 3 inputs function Butterfly diagram for Polarity 0



Negative polarity changes butterfly: polarity of x1 = 1, polarity of x2 = 0

$x1'x2 \oplus x1x2' = (1 \oplus x1)(1 \oplus x2') \oplus x1x2' = 1 \oplus x2' \oplus x1$

$\oplus x1\, x2' \oplus x1\, x2' = 1 \oplus x2' \oplus x1$



Problem that we want to solve

- Given is a Boolean function given as a vector of its minterms (true and false), a truth-table.
- Find one of 2^n FPRMs and its polarity for which the number of spectral coefficients is below some given cost bound (a number).

Complete example

$a'b'$ for polarity $a'b' = (00)$ ← FPRM polarity **Cost 1**

$a'b' = (1+a)b' = b' + ab'$ for polarity $ab' = (10)$ **Cost 2**

$a'b' = a'(1+b) = a' + a'b$ for polarity $a'b = (01)$ **Cost 2**

$a'b' = (1+a)(1+b) = 1 + a + b + a'b$ for polarity $ab = (11)$ **Cost 4**

polarity

Signal YES as a function of FPRM polarity and cost bound

For cost bound 1

a \ b	0	1
0	1	0
1	0	0

For cost bound 2

a \ b	0	1
0	1	1
1	1	0

For cost bound 3

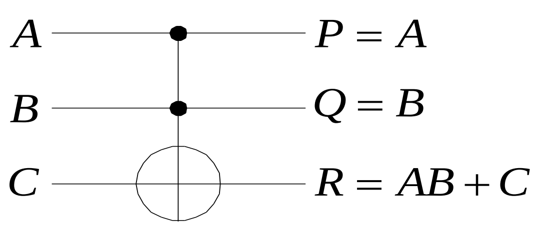
a \ b	0	1
0	1	1
1	1	0

For cost bound 4

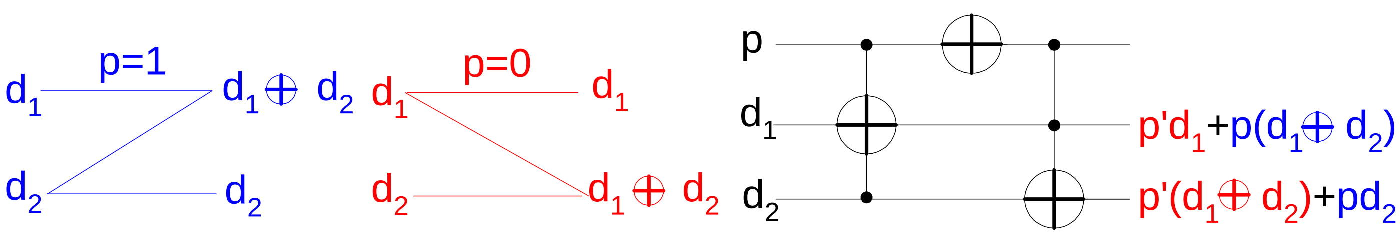
a \ b	0	1
0	1	1
1	1	1

R-M Butterflies Quantum Logic Circuit

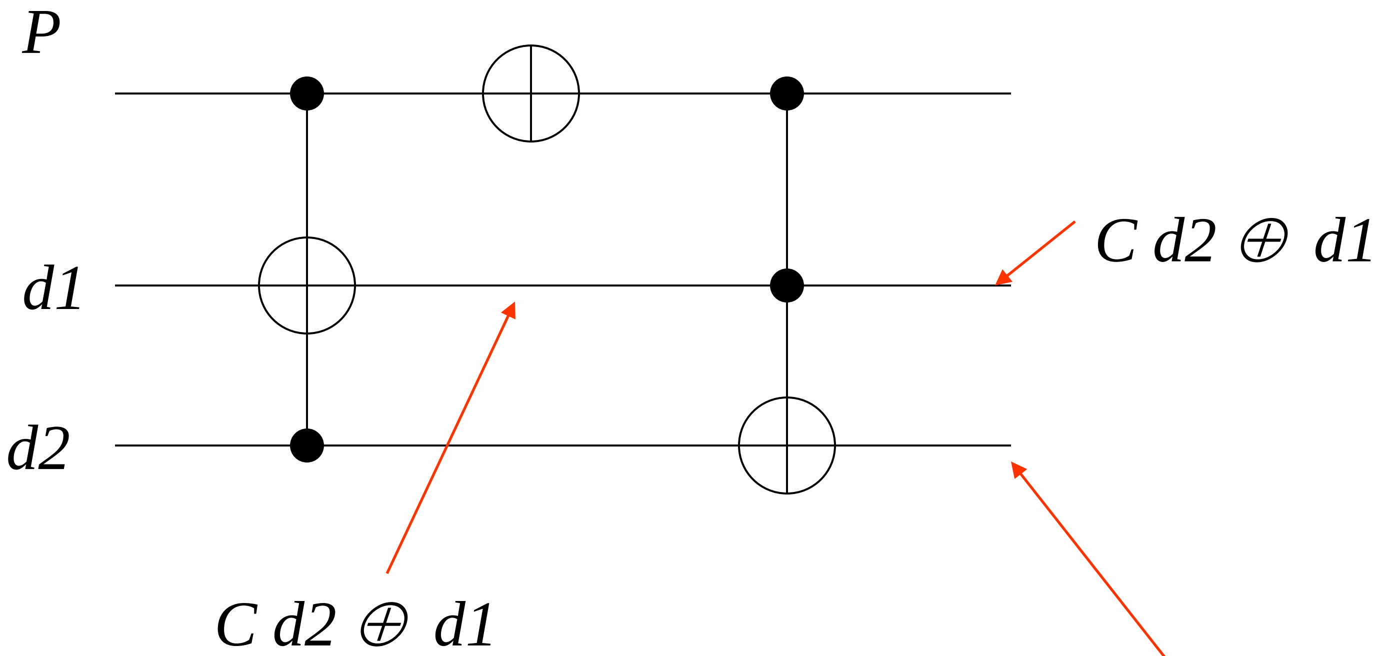
- A 3*3 Generalized Toffoli Gate



- Butterflies and corresponding Quantum circuit

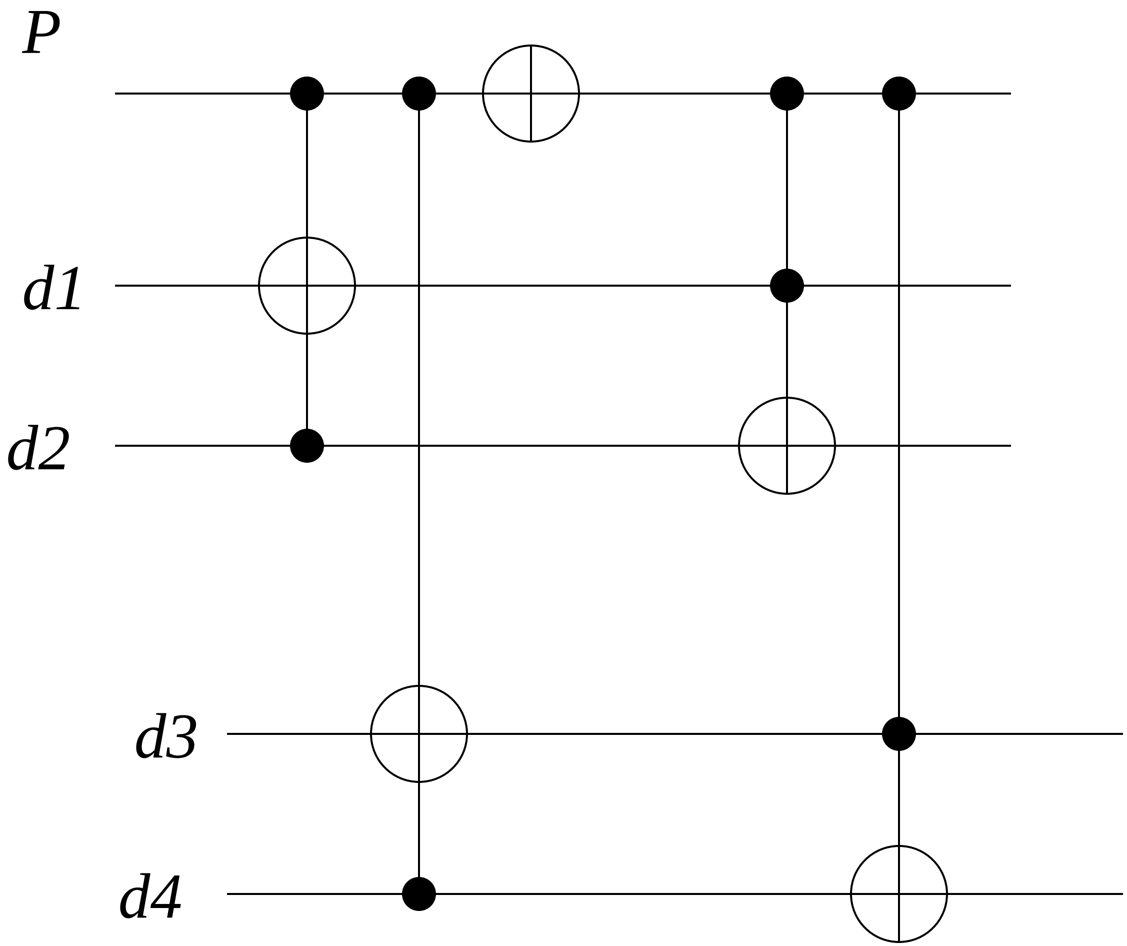


Quantum Kernel for FPRM



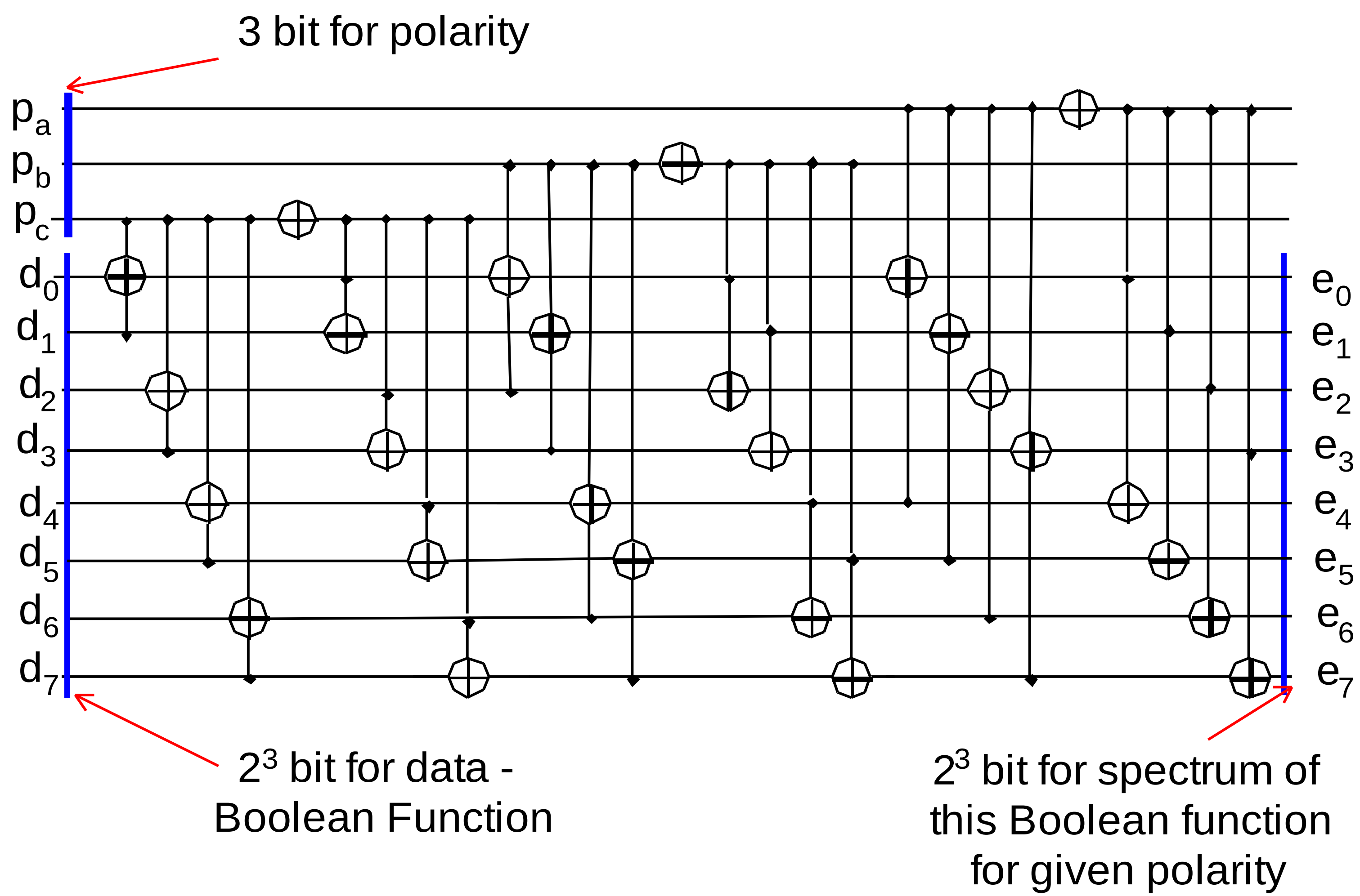
$$C'(C\,d2 \oplus d1) \oplus d2 = C'd1 \oplus d2$$

Quantum Data Path for FPRM



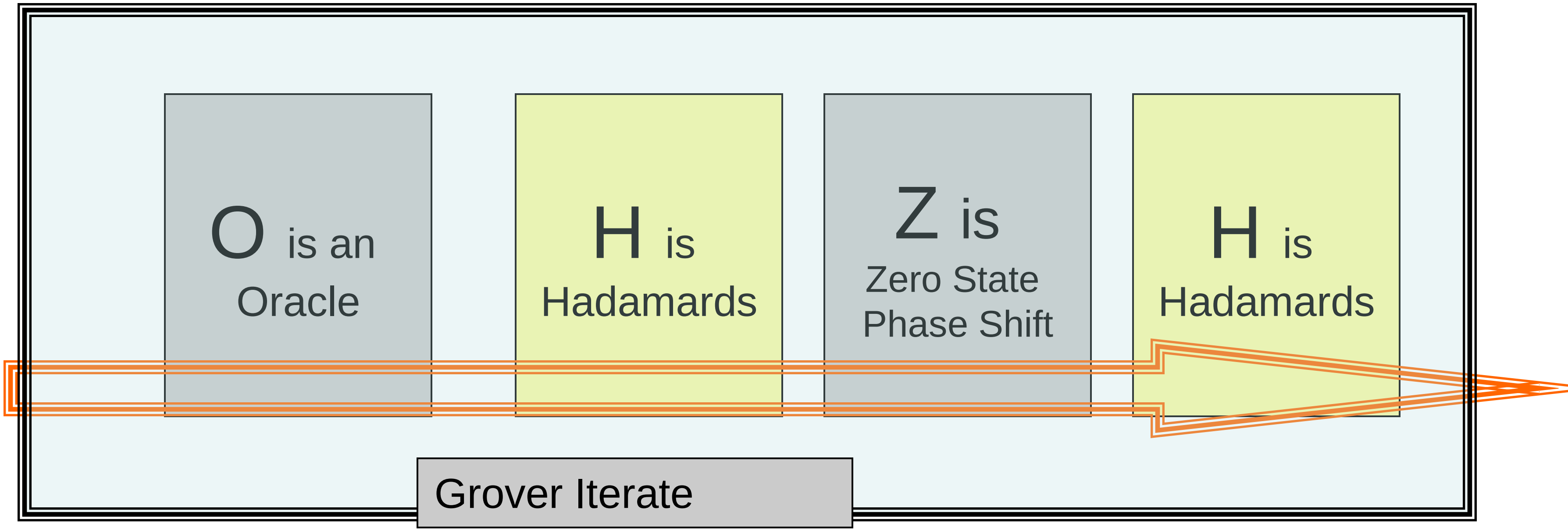
FPRM Processor

- Data path for all 3 variables FPRMs



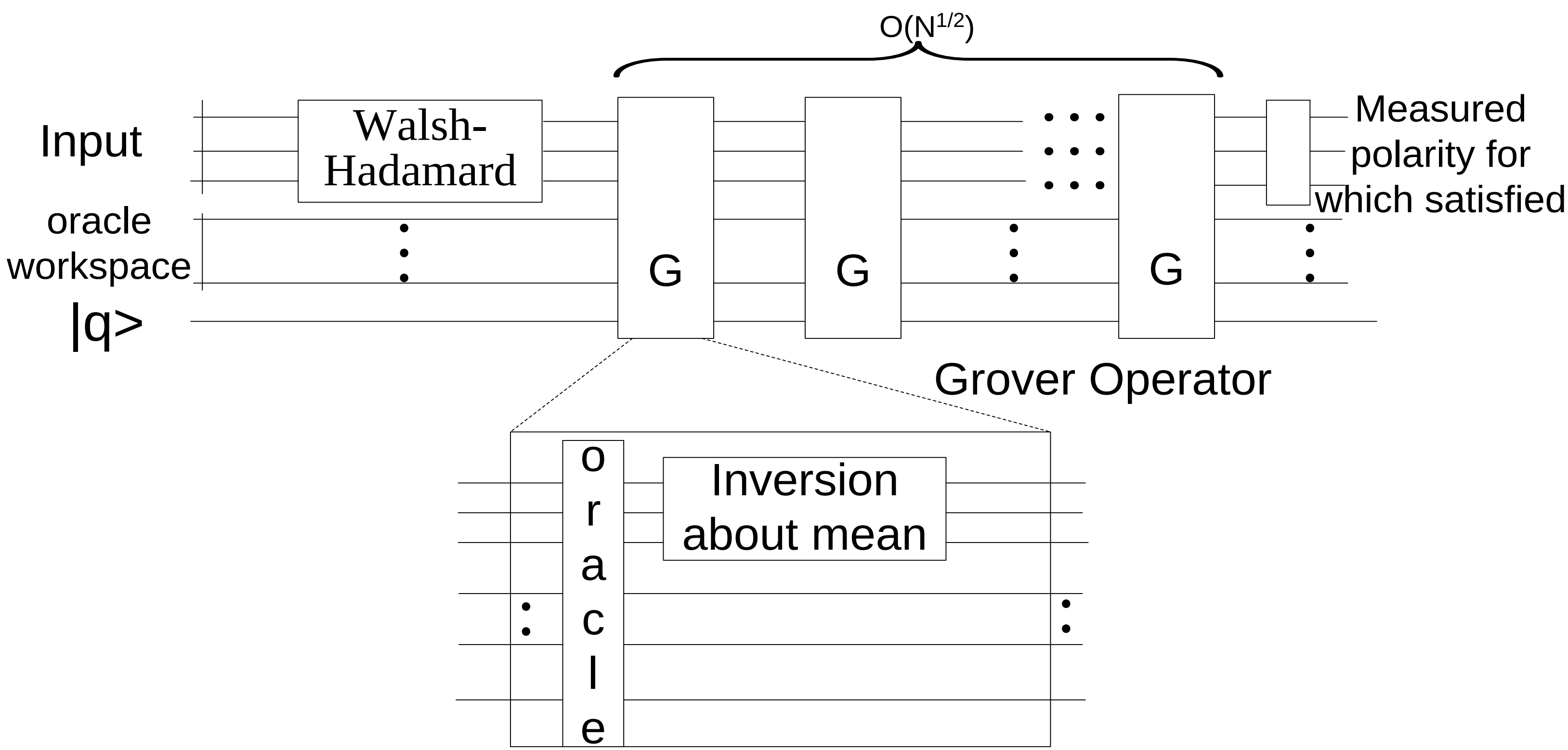
Components of Grover Loop (called also Grover Iterate)

- The Oracle -- O
- The Hadamard Transforms -- H
- The Zero State Phase Shift -- Z

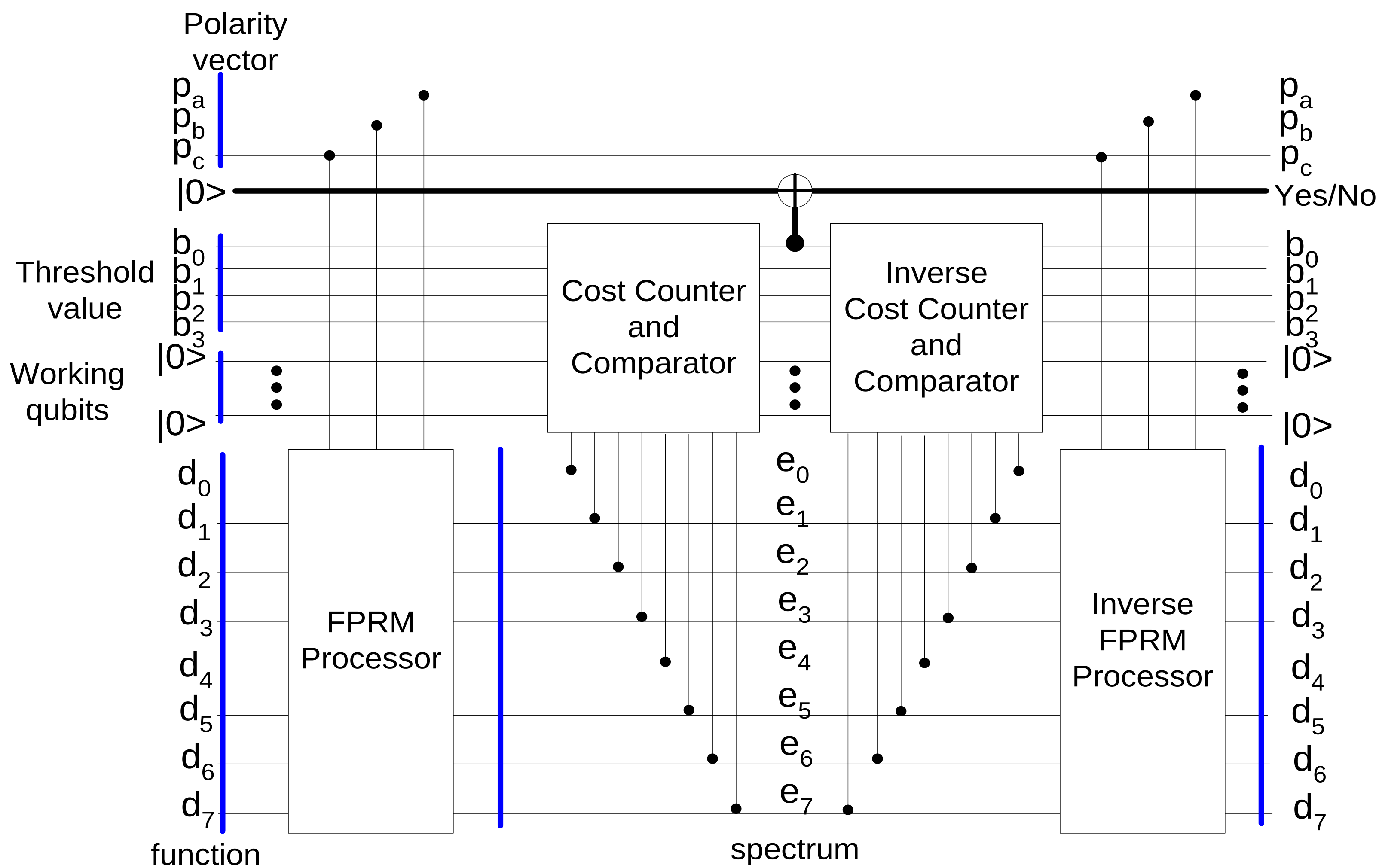


Grover's Algorithm

- 3 Steps for Grover algorithm
  - place a register in an equal superposition of all states
  - selectively invert the phase of the marked state
  - inversion about the mean operation a number of times

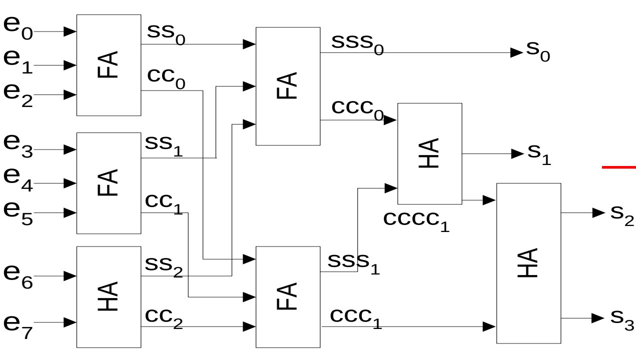


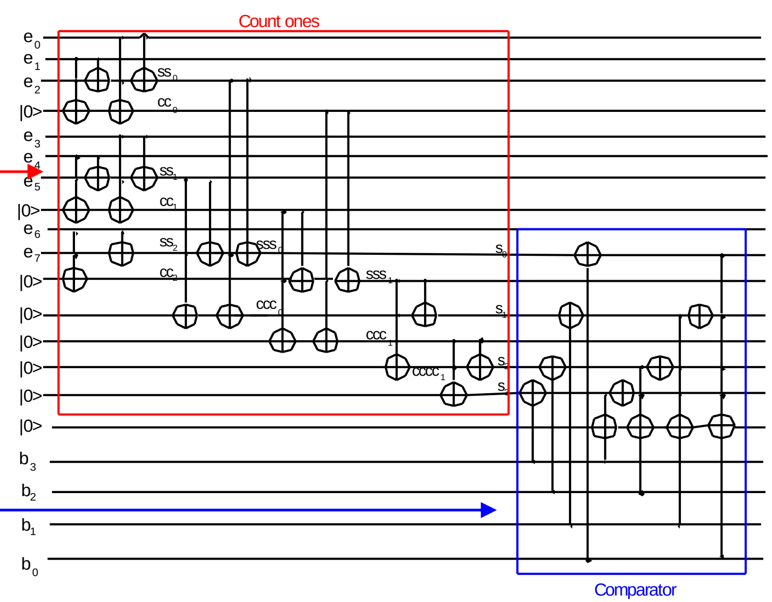
Quantum Architecture of FPRM oracle for Grover



Cost Counter and Comparator

- The first task is to count T ,
- The second task to evaluate the condition $P < T$.
- If the condition is true. the circuit will output one, otherwise zero.





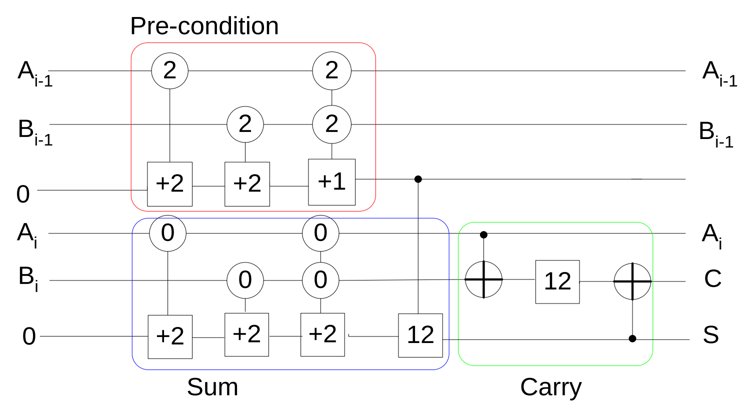
$$out = (s_3 \oplus b_3)b_3 \oplus (\overline{s_3 \oplus b_3})(s_2 \oplus b_2)b_2 \oplus (\overline{s_3 \oplus b_3})(\overline{s_2 \oplus b_2}) \bullet$$
$$(s_1 \oplus b_1)b_1 \oplus (\overline{s_3 \oplus b_3}) \bullet (\overline{s_2 \oplus b_2})(\overline{s_1 \oplus b_1})(s_0 \oplus b_0)b_0$$

MVL Compressor Tree Implementation

- More compact if using MVL compressor tree for cost counter and comparator
- Sign-bit adder and its quantum implementation

Table 1: Signed Binary Addition Table

$a_i + b_i$	Sign info. of digits in pos. i-1	c_{i+1}	s_{i+1}
$\bar{1}+\bar{1}$	Not Used	$\bar{1}$	0
$\bar{1}+0$	Either is Neg.	$\bar{1}$	1
$\bar{1}+0$	Neither is Neg.	0	$\bar{1}$
$0+0$	Not Used	0	0
$1+\bar{1}$	Not Used	0	0
$1+0$	Either is Neg.	0	1
$1+0$	Neither is Neg.	1	$\bar{1}$
$1+1$	Not Used	1	0



Other Problems that we solved with variants of this architecture

- **Problem 1.** Given is function and bound on cost. Find the FPRM polarity for which the cost of spectrum is below the bound.

- **Problem 2.** Given is polarity and bound on cost. Find the function such that FPRM in this polarity has the cost of spectrum that is below the bound.

- **Problem 3.** Given is polarity and function. Find the bound such that this function in this FPRM polarity has the cost of spectrum that is below the bound.

Essence of logic synthesis approach to Machine Learning

What oracle knows?

Description of Oracle criteria to separate beautiful from not beautiful

Angelina

Is Angelina beautiful?

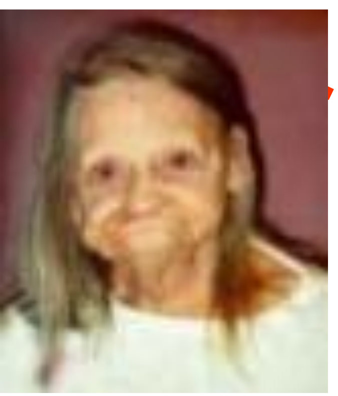
Oracle

yes

Oracle

no

Alice



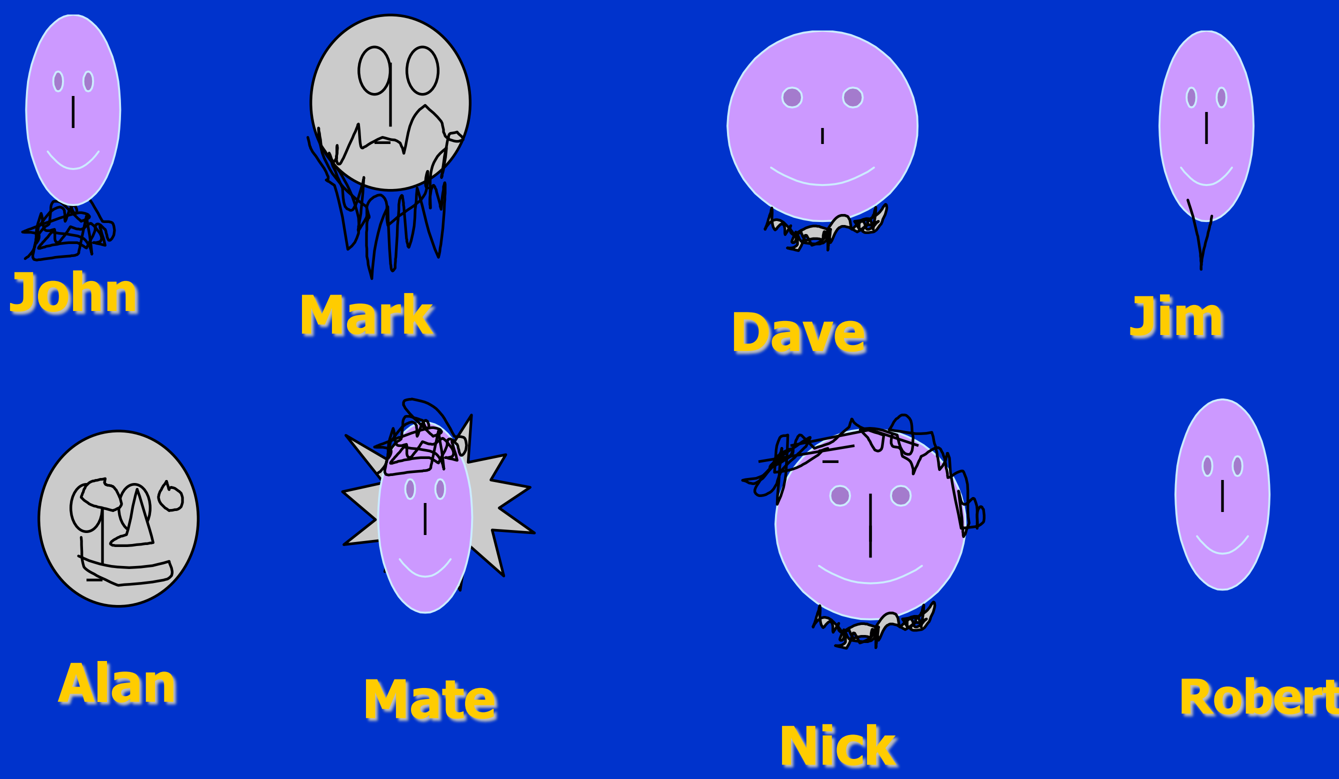
Is Alice beautiful?

observer

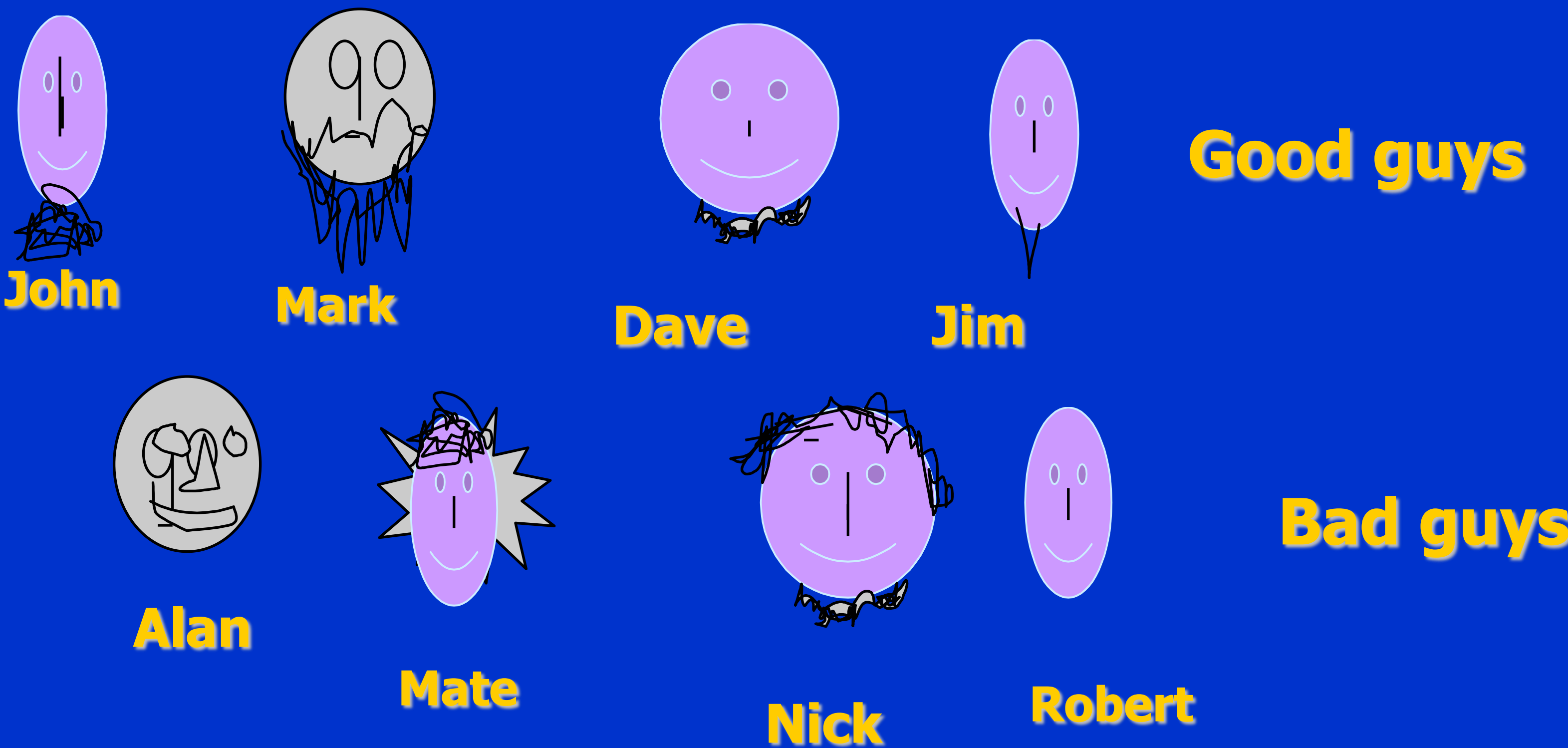
We have to learn oracle from examples

Example of Logical Synthesis for oracle creation

Who are the good guys?



Who are the good guys?

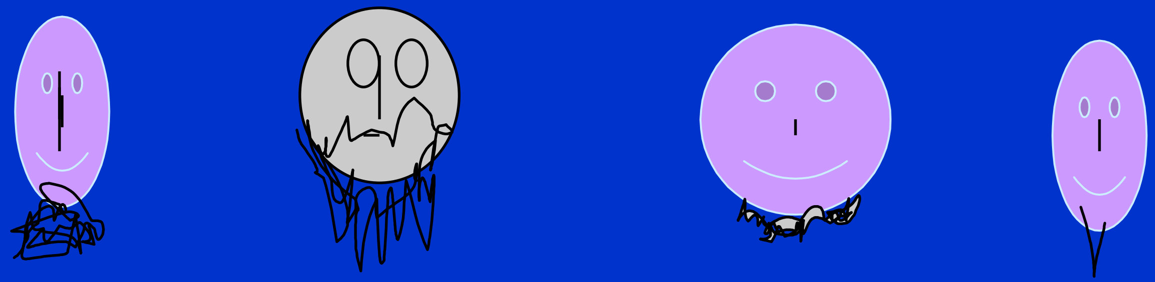


A - size of hair

B - size of nose

C - size of beard

D - color of eyes



John Mark Dave Jim

Good guys

A′ BCD   A′ BCD′   A′ B′CD   A′ B′CD

A - size of hair

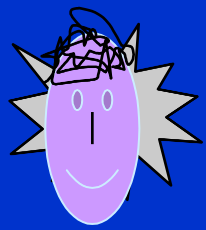
B - size of nose

C - size of beard

D - color of eyes

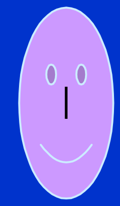
AB \ CD	00	01	11	10
00	-	-	1	-
01	—	-	1	1
11	-	—	-	-
10	-	-	-	-

Alan



Mate

Nick



Robert

Bad guys

A′ BC′D′ AB′C′D ABCD A′ B′C′D

A - size of hair

B - size of nose

C - size of beard

D - color of eyes

AB \ CD	00	01	11	10
00	-	-	1	-
01	0	0	1	1
11	-	—	0	-
10	-	0	-	-

A′C

Generalization 1:

Bald guys with beards are good

Generalization 2:

All other guys are no good

AB \ CD	00	01	11	10
00	-	-	1	-
01	0	0	1	1
11	-	—	0	-
10	-	0	-	-

A - size of hair

B - size of nose

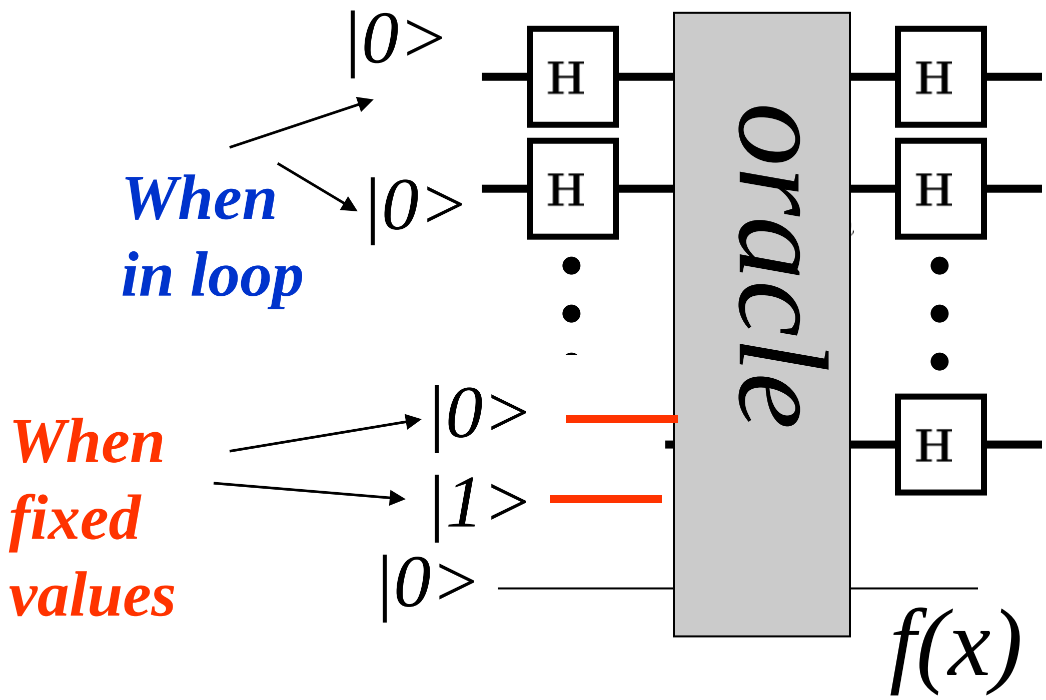
C - size of beard

D - color of eyes

A'C

Other Problems that we solved with variants of this architecture

- **<u>Problem 4.</u>** Given is an incompletely specified function. Find the FPRM polarity for which the cost of spectrum is the minimum.



This is the machine learning problem just shown

Other Applications

- Logic Design
  - (also logic minimization for reversible and quantum circuits themselves)
- Image Processing
- DSP

Applications

- Quantum Game Theory.
  - For instance, the problem discussed above is more general than the game of finding the conjunctive formula of literals for a given set of data.

Applications

- All circuits presented here can be generalized to ternary quantum gates, allowing for ternary butterflies and more efficient arithmetic for larger counters and comparators.

Conclusion

Hi guys, you just learnt a method that allows everybody who knows how to design a reversible oracle to create a Grover-based quantum algorithm for a new NP-hard problem