Inverse Problems, Constraint Satisfaction, Reversible Logic, Invertible Logic and Grover Quantum Oracles for Practical Problems

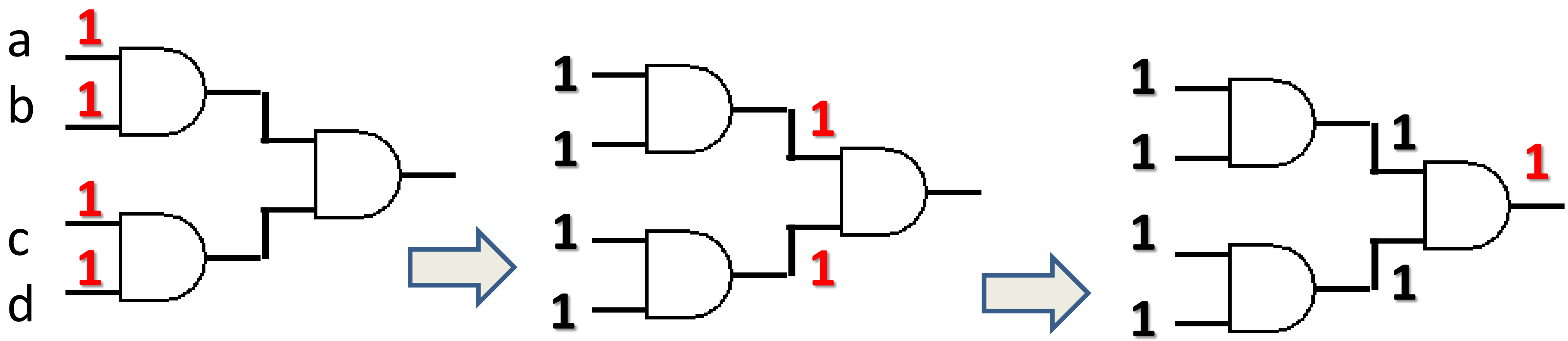
Marek Perkowski
Department of Electrical and Computer Engineering
Portland State University

Friday, July 10th 2020

What is an Oracle?

- Technical aspects can be found in the paper and its references
- Here I want to concentrate on fundamental ideas and our philosophy
- So that everybody can start using these ideas in their related areas of research

Classical Boolean Circuits propagate signals only from inputs to outputs



What is an Oracle in Quantum Computing?

A Boolean function ***with one output*** we will call an Oracle

Oracle is a fundamental concept in Grover Algorithm, which is the famous quantum algorithm with many applications.

Propagating signal through a combinational circuit from input to output is called evaluation of the oracle.

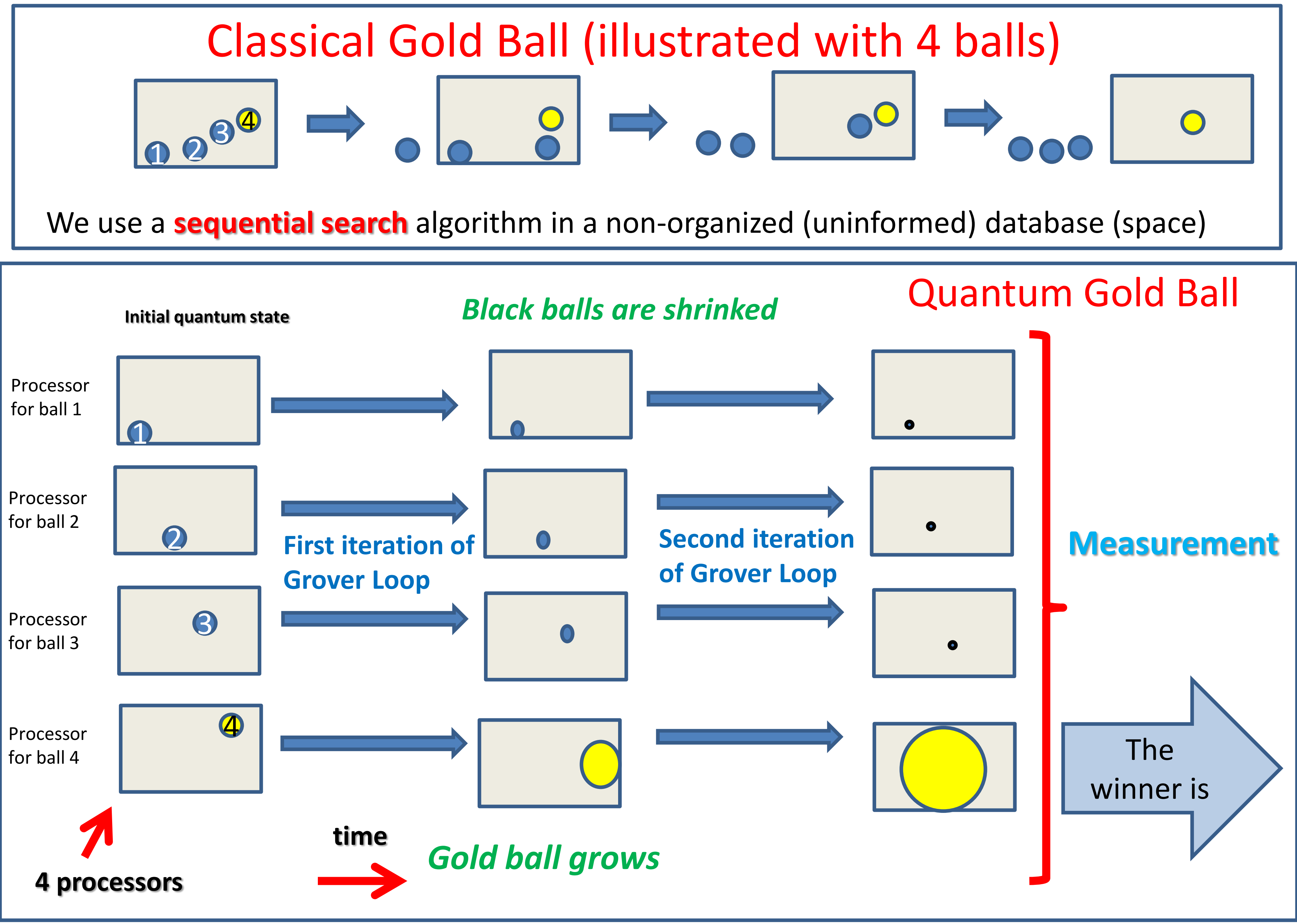
With input combination a=1,b=1,c=1,d=1 we need to propagate signal only once through this oracle to obtain value "1" on output.

The Essence of Quantum Algorithm of Grover

A gold ball in a box or a miracle of a quantum algorithm

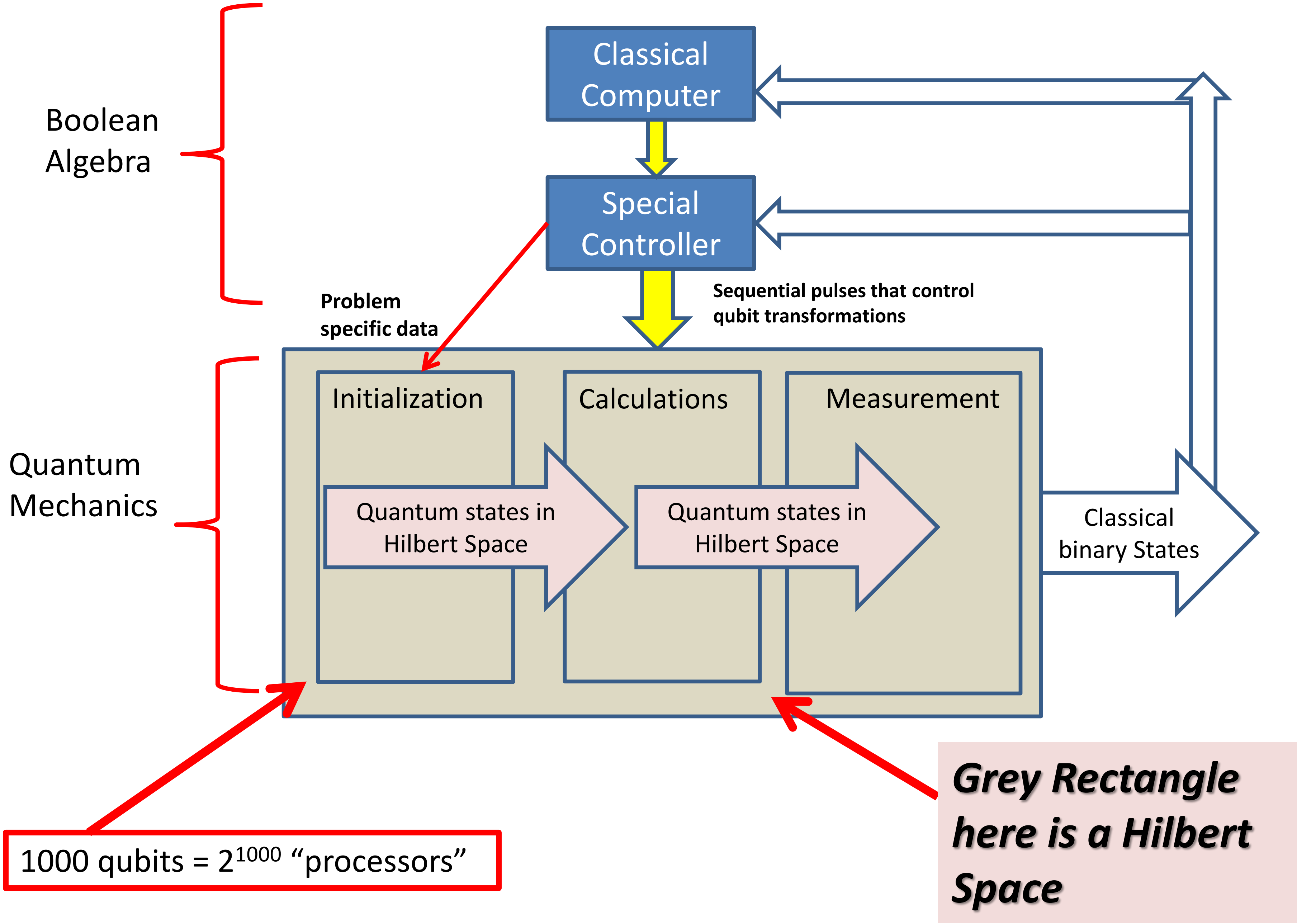
- We have one **gold ball** in a **black box** and 99 black balls.
- We do not know anything about what is inside the box, other that there is a gold ball inside.
- We reach to the box, if the ball is gold we are happy, otherwise we remove the black ball from the box and reach again.
- We repeat this until we find the gold ball.
- In the best case we find the ball in the first attempt, in the worst case - in the attempt number 100.
- On average we need to reach 50 times to the box.
- In quantum we need to reach only $\sqrt{100}$ =10 times to find the gold ball.
- **Why it is so**?

The answer is quantum superposition or quantum parallelism



We use a **parallel search** algorithm using superposition in Hilbert Space

Modern Quantum Computer is a Hybrid Machine



Conclusion on Grover

- Power of quantum computing is in very high parallelism.
- Having 2 particles is equivalent to having 4 computers.
- Having n particles means that we have a parallel computer with 2^n processors.
- Now there are universal quantum computers with 100 qubits (particles).
- This means 2^{100} processors working in parallel.
- But this parallelism is not for all problems.

Let us go back to oracles

a,b\c,d	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	0	0	**1**	0
10	0	0	0	0

False minterm

True minterm

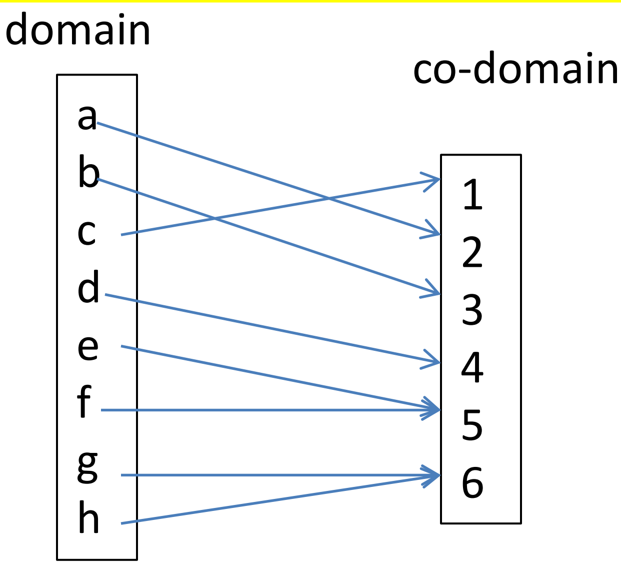
Boolean Function formulation of the Gold Ball problem

- Now the gold ball is a "1", the black balls are the "0".
- I want to find for which combination of input a,b,c,d the output is 1.
- I do not know what is the function, I can only guess the inputs and check the output.
- This is like a testing problem.
- I give one minterm at a time.
- In the worst case I need to test 16 minterms.
- In the best case I need to test 1 minterm, on average I need 8 minterms.
- Grover will find the solution in $\sqrt{16}$ = 4 tests.

1. Assume that the oracle circuit of function F2 is just a tree of two-input AND gates with 64 inputs in total.
2. To find the 1 of function, the oracle in classical logic would be evaluated 2^{64} number of times.
3. Grover Algorithm would evaluate the oracle "only" 2^ (64/2) = 2^{32} times which may be also not practical.
4. However, in Invertible logic in which one propagates the signals from output to inputs, the **Invertible Logic** method would need only one evaluation of the oracle realized with invertible gates

Inverse Problems

Inverse Problems for functions

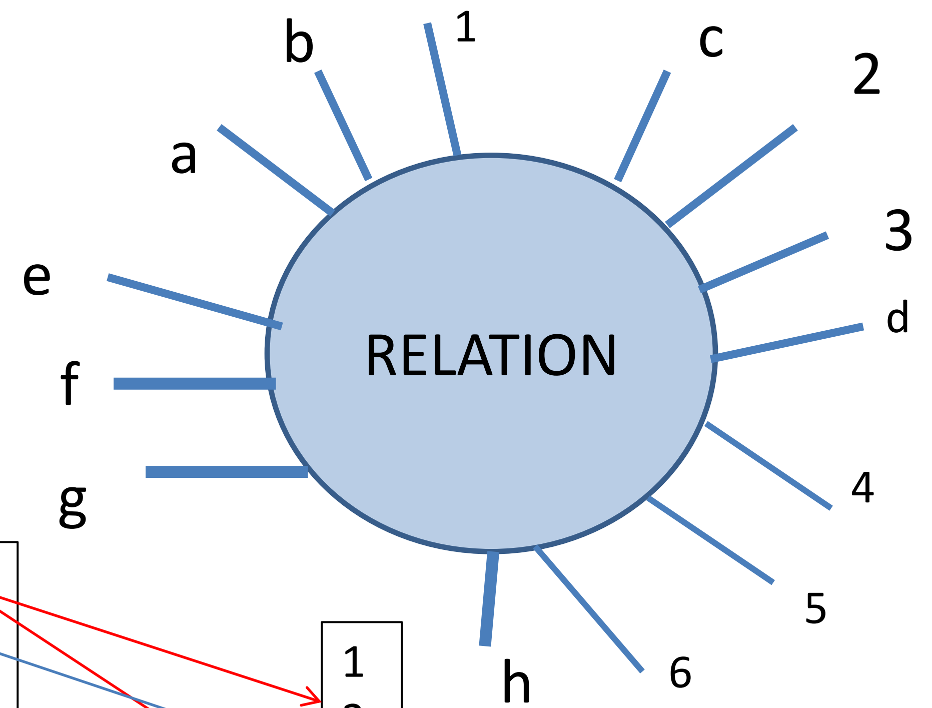


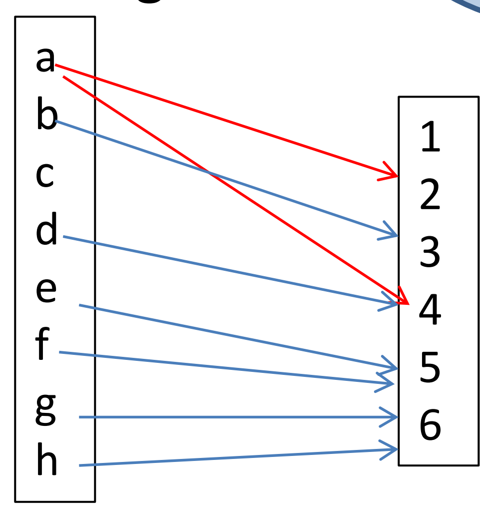
What are all letters that map to number *k* ?

Characteristic function

CNOT gate				
Input		Output		
D0	D1	D0	D1	F =
0	0	0	0	1
0	1	0	1	1
1	0	1	1	1
1	1	1	0	1

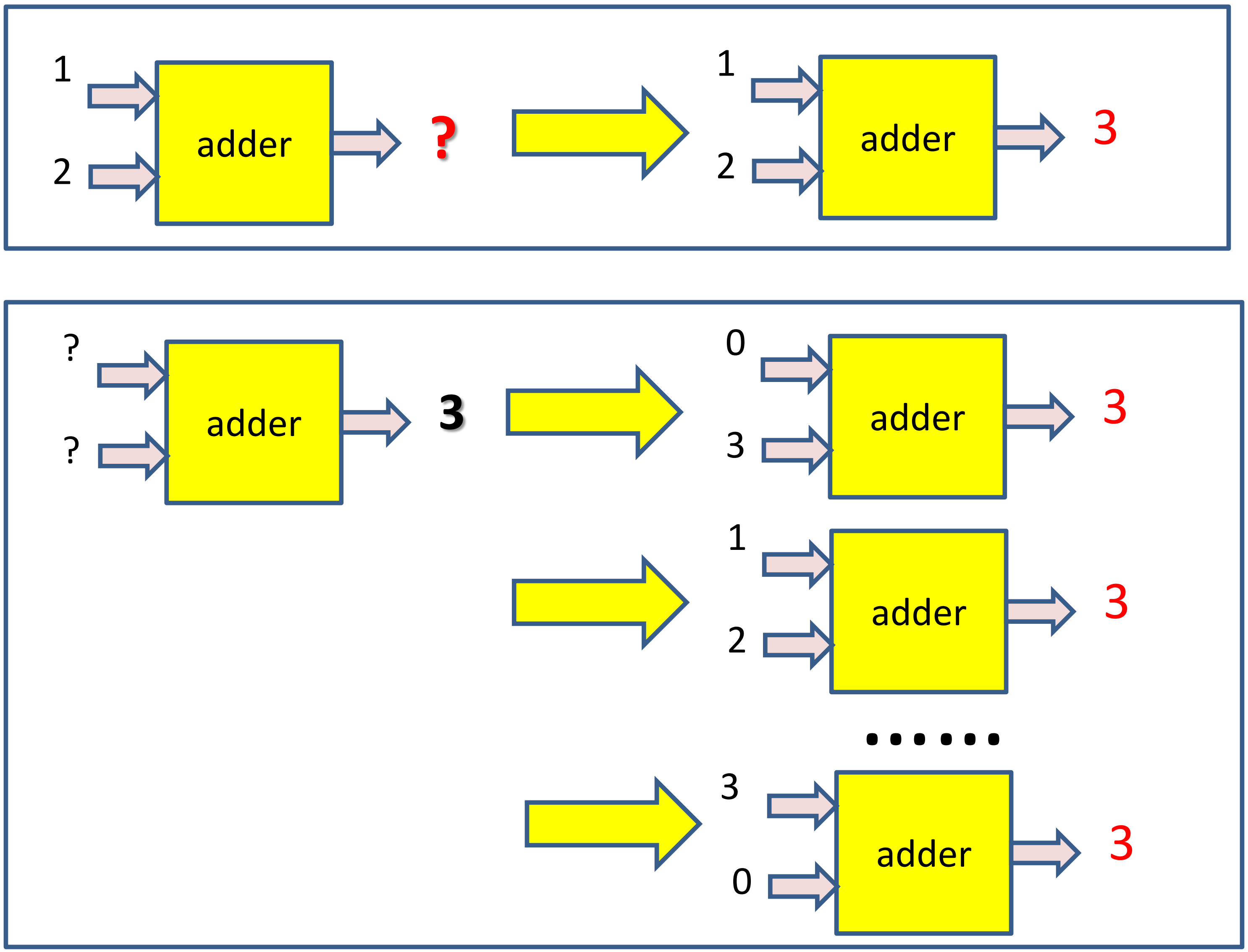
Inverse Problems for relations





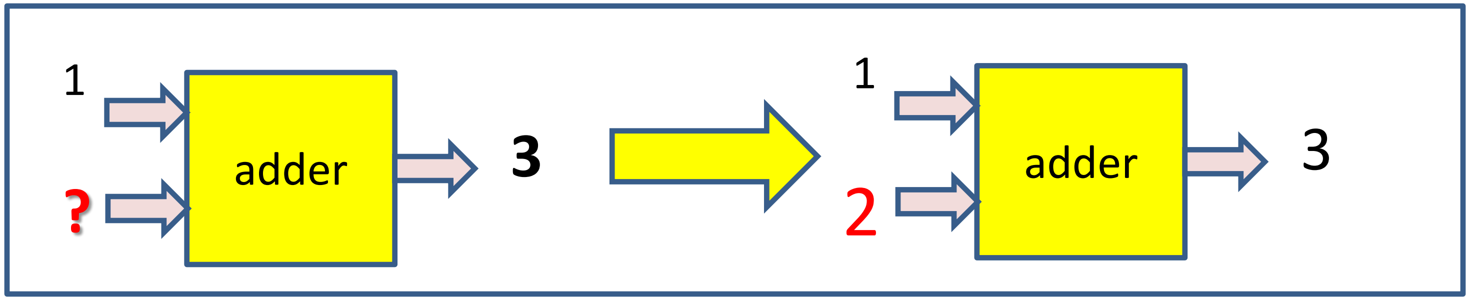
- What are all letters that map to number *k* ?
- Is characteristic function satisfied?

Examples of Inverse Problems solved by Invertible Logic

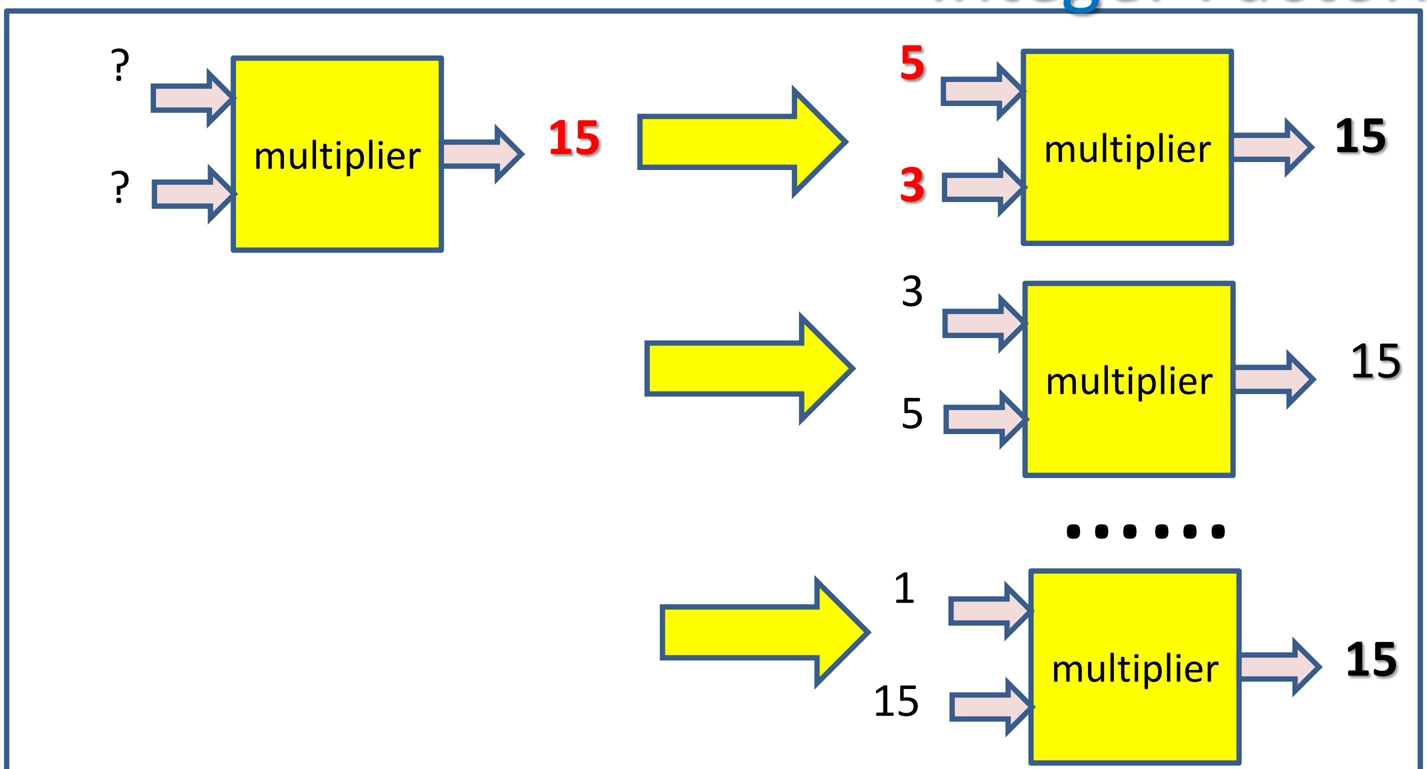


… but also…. Generalized Inverse Problems

Signal propagation in all directions



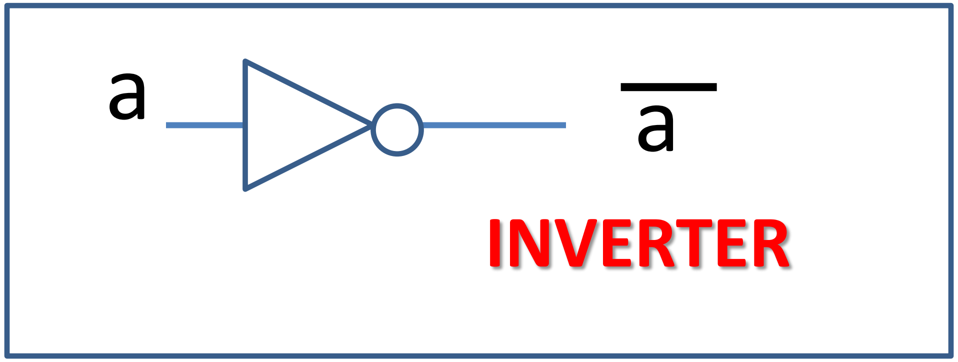
Integer Factorization

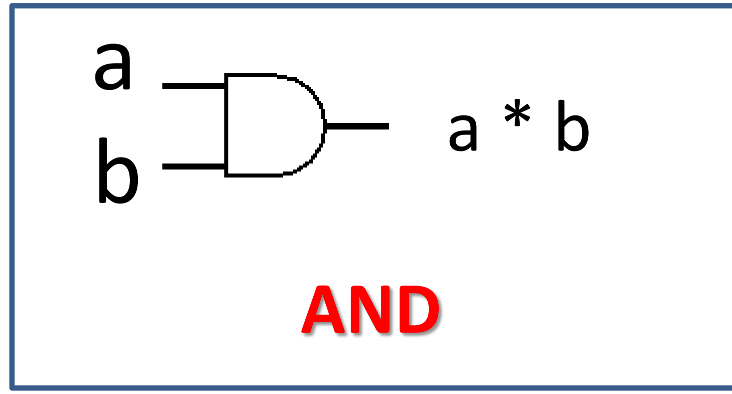


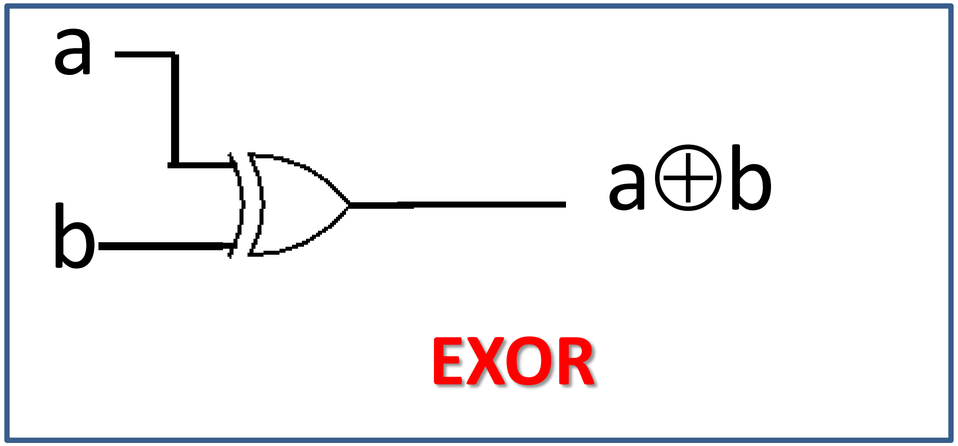
1. Cryptography
2. Quantum cryptography
3. Post-quantum cryptography

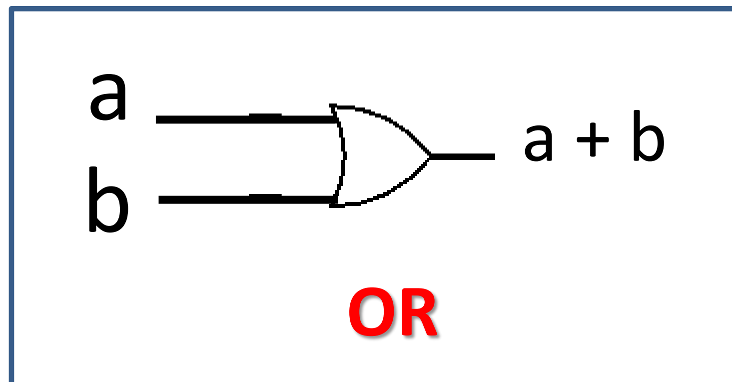
Reversible Logic

Notation for Classical Boolean Logic Gates and **Invertible** Logic Gates



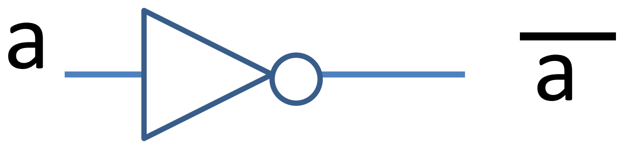




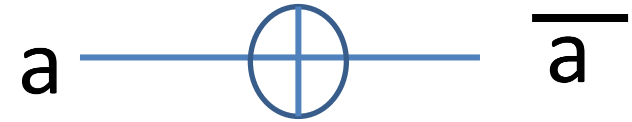


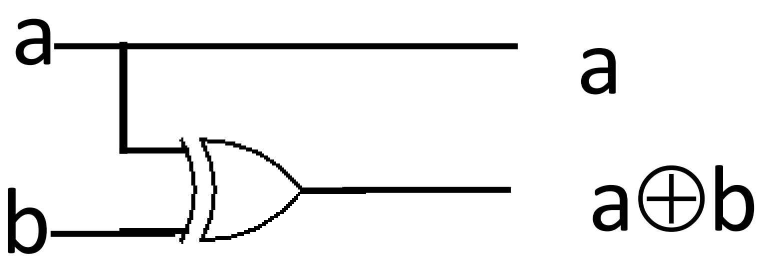
These gates propagate from inputs to outputs

Notation for Reversible Logic Gates

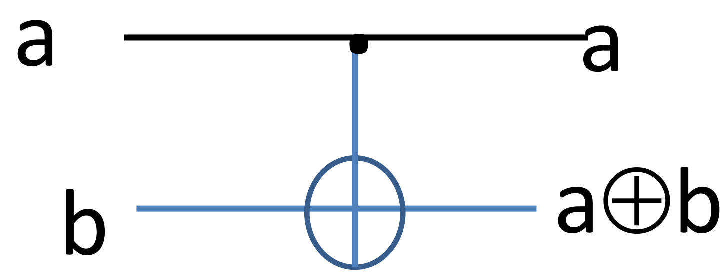


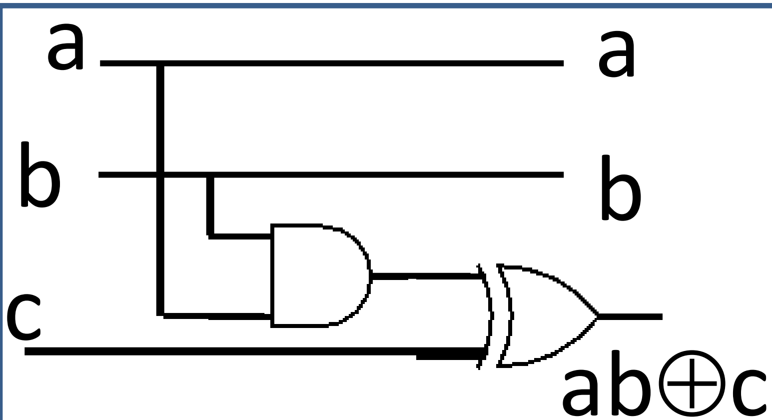
Inverter



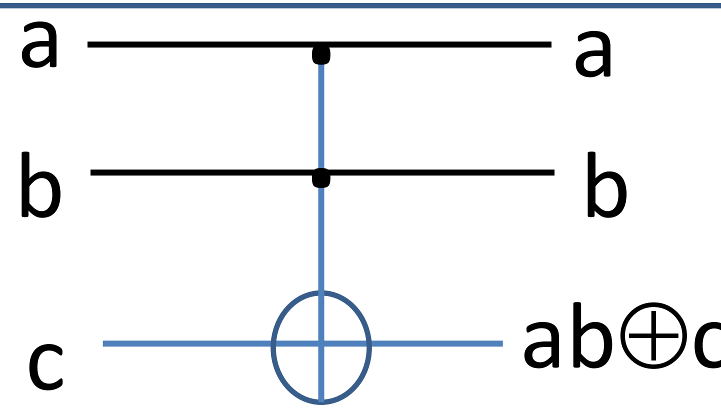


Controlled-NOT or Feynman Gate





Controlled-Controlled-NOT or Toffoli Gate



These gates are in reversible logic. Propagate from input to output or from output to input.

In quantum realization they allow for superposition and entanglement

Reversible Circuits from Reversible Gates

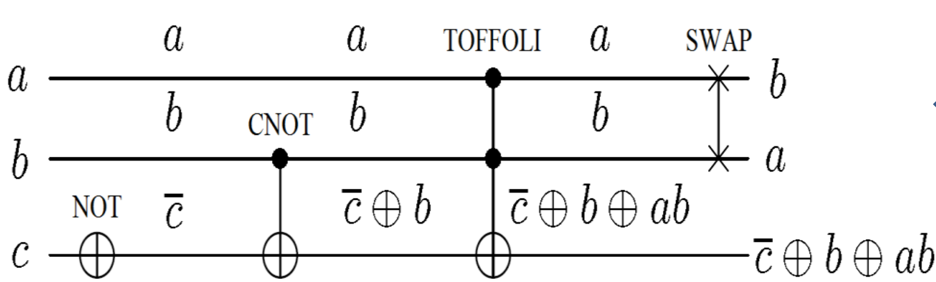
Truth Tables

NOT gate		CNOT gate				TOFFOLI gate						SWAP gate			
Input	Output	Input		Output		Input			Output			Input		Output	
D0	D0	D0	D1	D0	D1	D0	D1	D2	D0	D1	D2	D0	D1	D0	D1
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	1	0	1	0	0	1	0	0	1	0	1	1	0
		1	0	1	1	0	1	0	0	1	0	1	0	0	1
		1	1	1	0	0	1	1	0	1	1	1	1	1	1
						1	0	0	1	0	0				
						1	0	1	1	0	1				
						1	1	0	1	1	1				
						1	1	1	1	1	0				

These are reversible functions, one-to-one mappings or permutations.

These are also reversible gates from which reversible circuits are composed.

Schematic Diagram

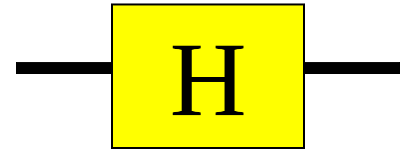


- This is an example of a circuit created from reversible gates without ancilla bits.
- The corresponding function is reversible.

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

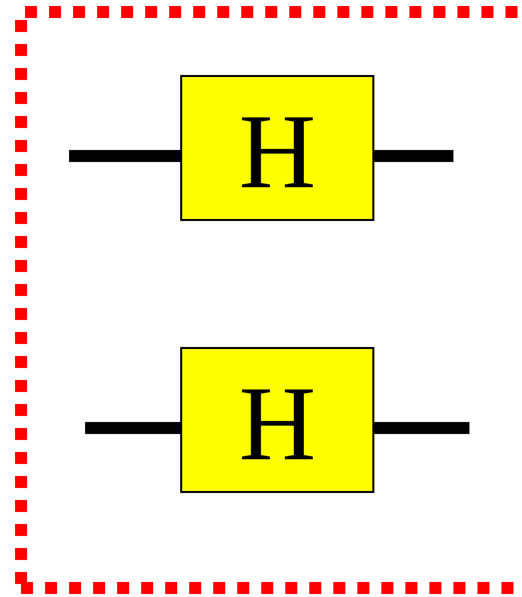


$$\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} \otimes \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix} =$$

= 1/2

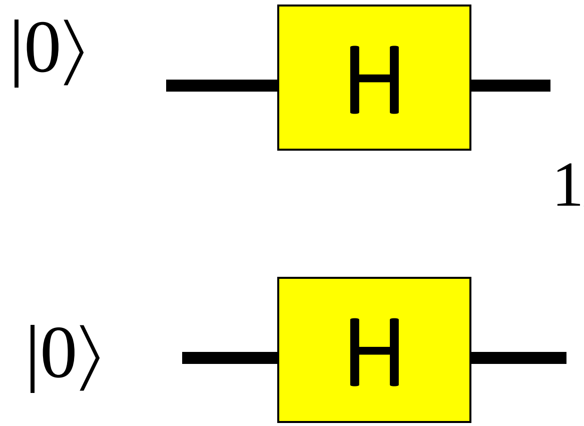
1	1	1	1
1	-1	1	-1
1	1	-1	-1
1	-1	-1	1

Here I calculated Kronecker product of two Hadamards



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

Hadamard Transform



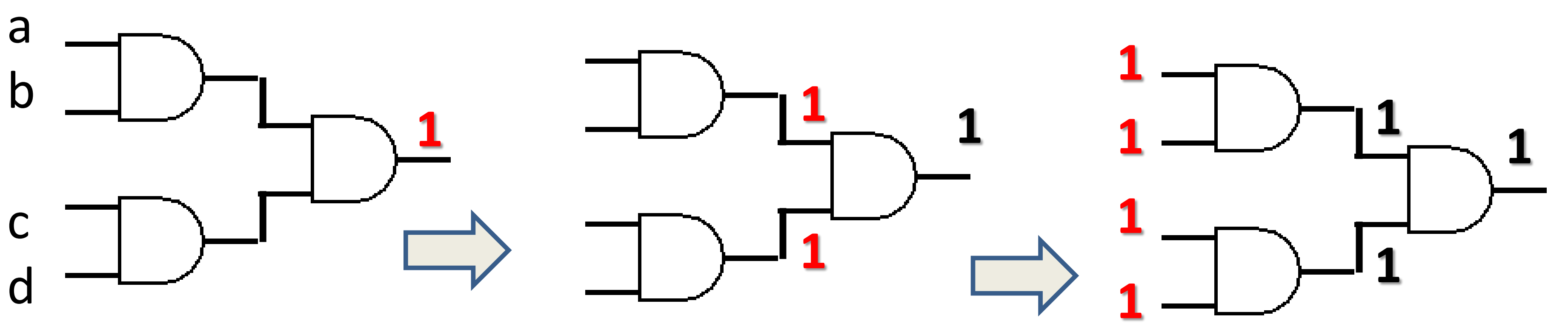
$$\frac{1}{2}|00\rangle + \frac{1}{2}|01\rangle + \frac{1}{2}|10\rangle + \frac{1}{2}|11\rangle$$

After vector of Hadamards we have equal superposition of all states from my problem space

Invertible Logic

What is Invertible Logic?

1. Invertible Logic algebraically is exactly the same as classical Boolean Logic.
2. The only difference is that the invertible logic CIRCUIT can just propagate signals:
   1. From inputs to outputs
   2. From outputs to inputs
   3. From any subset of inputs and outputs and internal signals in any direction.
3. Invented by **Professor Datta from Purdue** as a result of investigation of Deep Recursive Neural Nets.



So now, in ***only one*** evaluation of oracle we find the solution a=1,b=1, c=1,d=1. **F = abcd.**

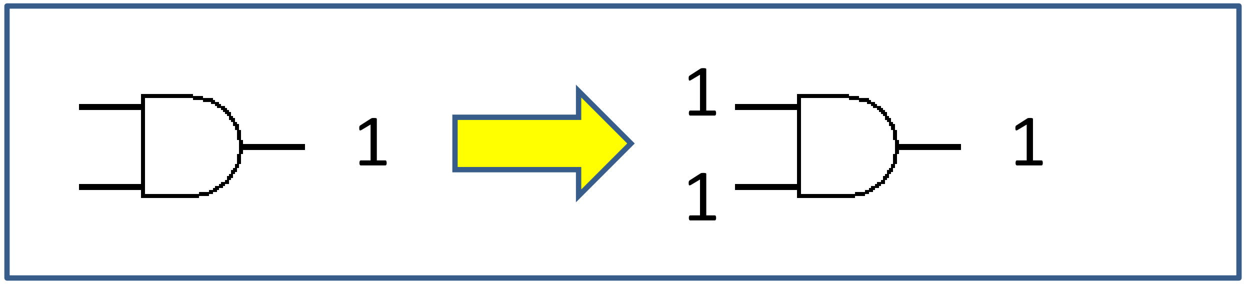
This is an example of Inverse Problem

- For n=64 variables the quantum Grover Algorithm will need to evaluate the Oracle 2^{32} times. **NOT GOOD**.
- Invertible Logic will find solution in **only one evaluation** of oracle.

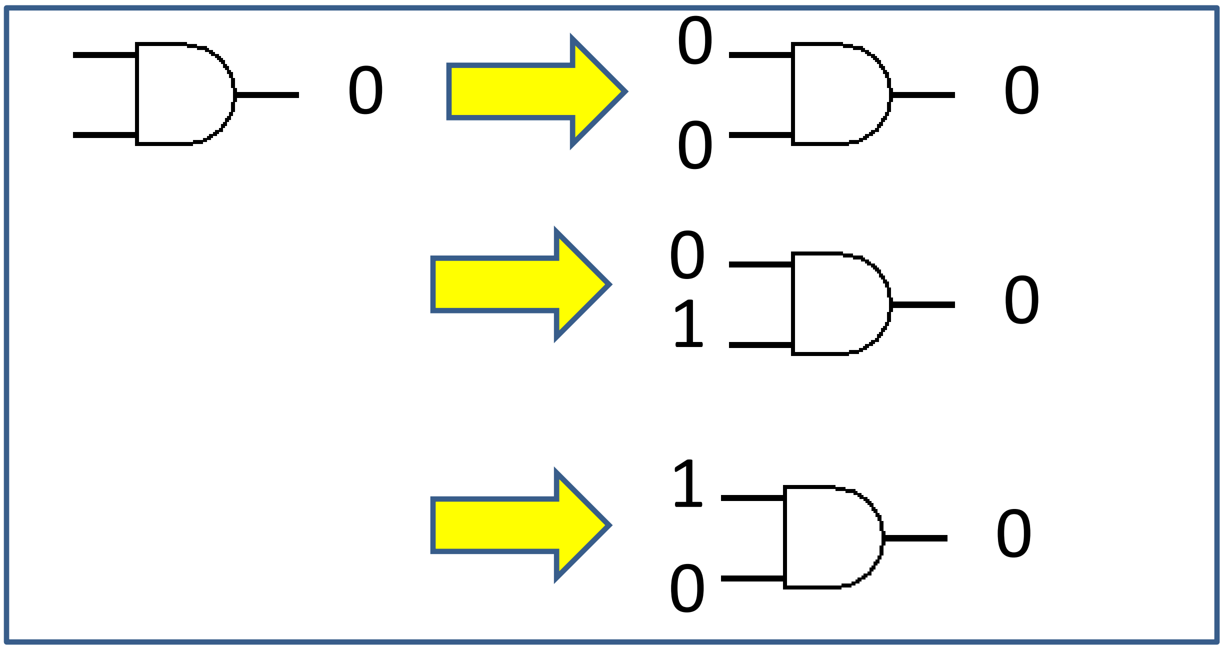
Few important points

- The creator of the oracle does not know the function like we can see it in a Karnaugh Map.
- He can create the oracle as a circuit without knowing the function.
- This oracle can be evaluated using:
  a. A classical Boolean Oracle
  b. A Quantum Oracle
  c. An Invertible Logic Oracle.

How Invertible Logic works?



Deterministic propagation



Non-deterministic propagation

Similarly we can analyze any Boolean Gate or block such as adder

How Invertible Logic works?

A circuit is a tree of Boolean Gates $F_2 = (ab) \oplus (cd)$ as in Figure 16. The snapshots show the propagation of signals backward with fast finding of one solution. EXOR is a better combining gate than the OR gate, because for output 1 it has only two not three input combinations (0,1) and (1,0).

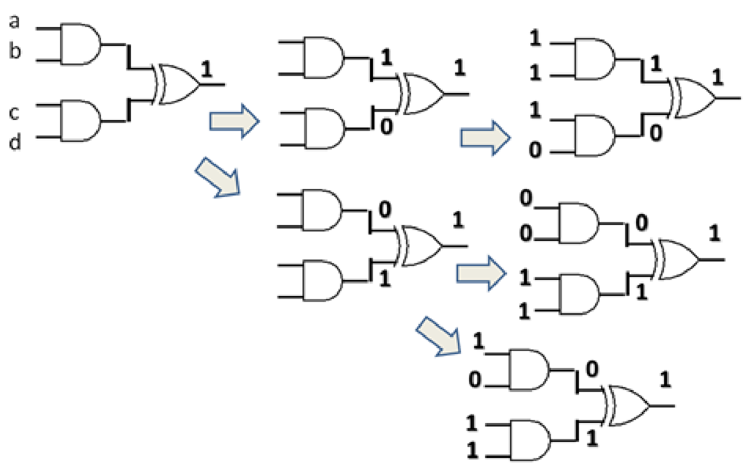


Fig. 16

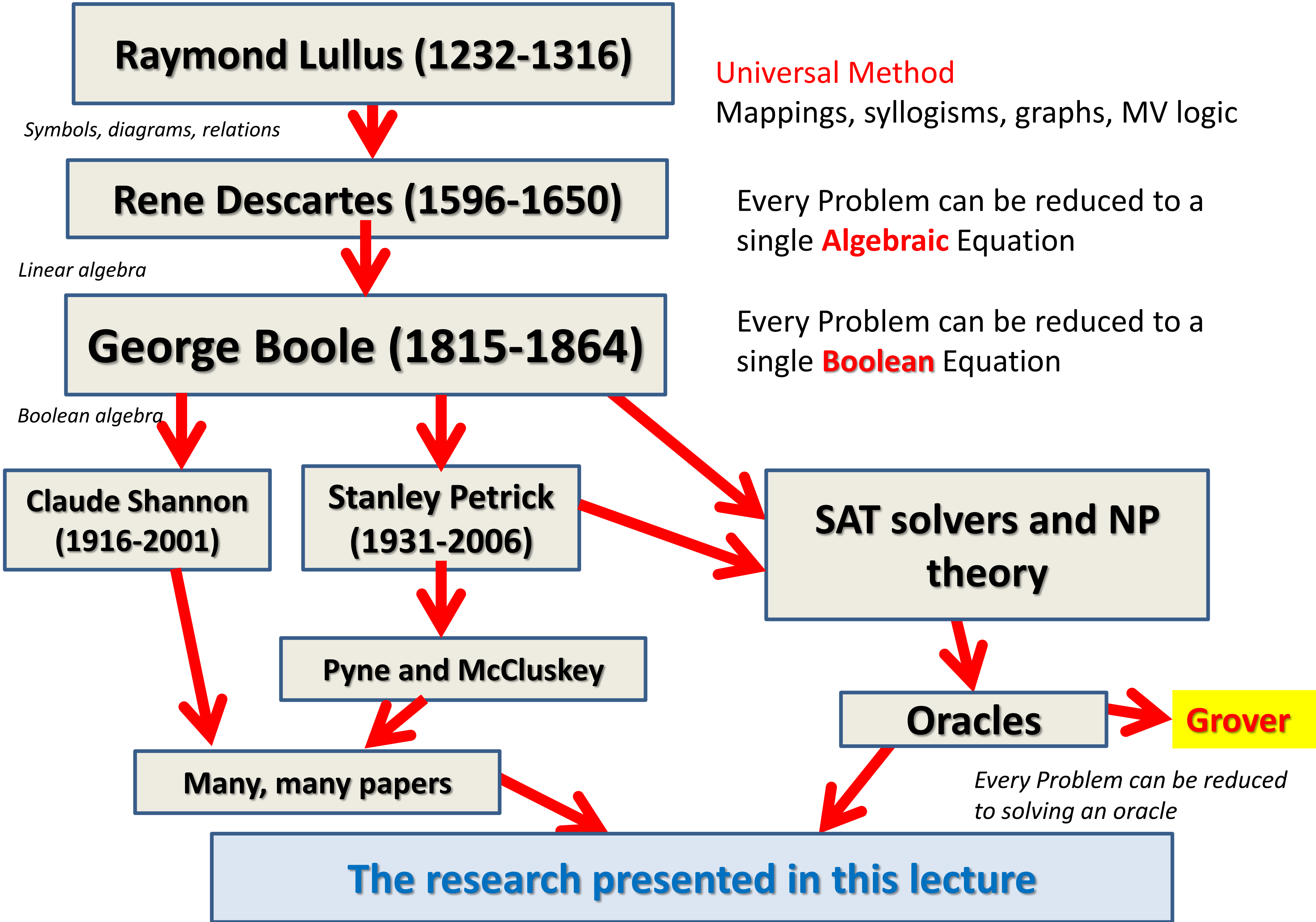
- **Invertible Logic solves Inverse Problems.**
  - **Graph Coloring**
  - **Integer Factorization**
  - **Cryptology**
  - **Bitcoin**
  - **Puzzles**
  - **Electronic Design Automation**
  - **Inverse Kinematics**

And every problem described by an oracle

- **Invertible Logic is a general concept**
- ***It is in principle not related to any technology***

A Dream of Universal Method “to Solve All Problems”

Old History of Universal Method for Problem Solving



Universal Method
Mappings, syllogisms, graphs, MV logic

Every Problem can be reduced to a single **Algebraic** Equation

Every Problem can be reduced to a single **Boolean** Equation

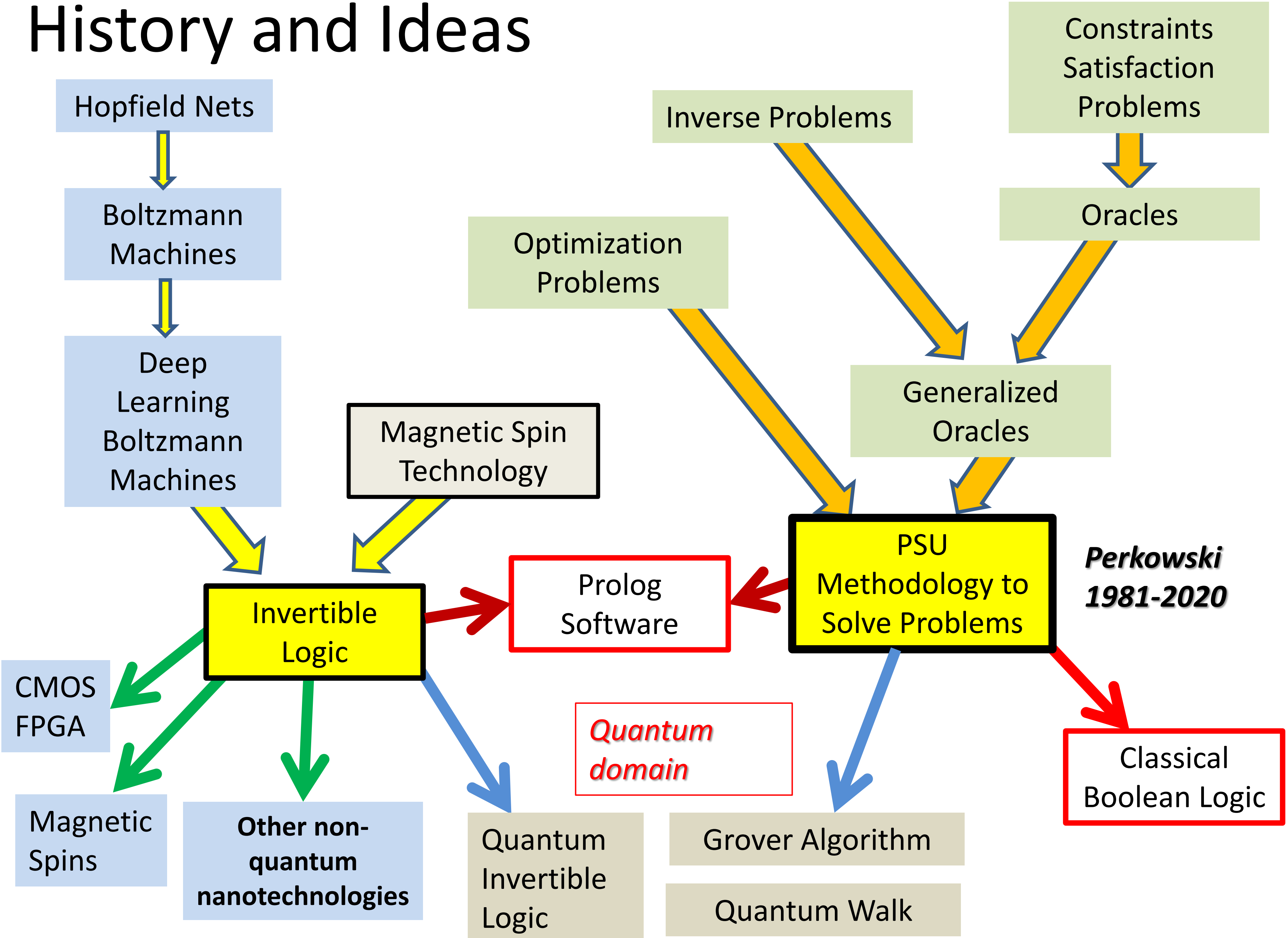
Every Problem can be reduced to solving an oracle

George Boole said:

1. Describe a problem by a set of simple logic equations.
2. Convert the set of these equations to a single logic equation.
3. Solve this equation to get all answers.

"universal method" to solve logic problems

History and Ideas



What is my achievement?

- People who work on Invertible Logic do not have a methodology to develop oracles.
- We combine our methods of classical, reversible and invertible logic to build a very large class of oracles and reduce problems from various areas to building oracles.
- **Optimization Problems** are reduced to sequence of oracles with modified constraints.

Constraint Satisfaction Problems

Constraint Satisfaction

- **Given** is a set of arithmetic, Boolean, Predicate and other constraints on a set of variables.
- **Find** all vectors of values of variables that satisfy ***all constraints***

- **Given** is a set of arithmetic, Boolean, Predicate and other constraints on a set of variables.
- **Find** all vectors of values of variables that satisfy ***as many as possible constraints***

Optimization

- **Given**
  1. is a set of arithmetic, Boolean, Predicate and other constraints on a set of variables
  2. Cost function.
- **Find** all vectors of values of variables that satisfy ***all constraints*** **AND** ***optimize the cost function***

Constraint Satisfaction Problems

- Graph Coloring
- Maximum Clique
- Maximum Independent Set
- SAT and MAX-SAT
- 8 Queens
- Logic Puzzles
- Integer Linear Programming
- Partition Calculus Problems
- POS to SOP conversion
- FPRM Minimization

- Two-level logic minimization
- Finite State Machine Minimization
- FSM encoding
- Concurrent encoding and minimization of FSM
- Ashenhurst-Curtis Decomposition of Logic functions.
- Minimum Set of Support
- Rule Minimization
- Binate Covering
- Unate Covering
- Logic methods of Machine Learning.
- Generalized Traveling Salesman
- Generalized Knapsack
- Shortest Path and several robot planning tasks

- Test generation
- Cryptography
- Logistic problems
- Non-linear Algebraic equations

Perkowski

Yiwei Li

Sharavana Kumar, visitor from India

Wei Zhang, visitor from China

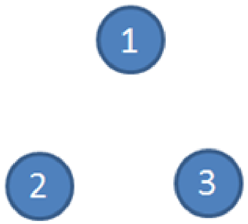
Peng Gao

Wenjun Hou, high school

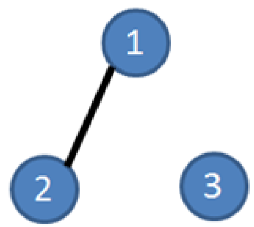
Jacob Viertel from Germany

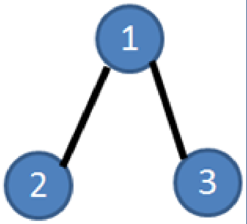
Example of CSP – Graph Coloring

This graph colored with one color

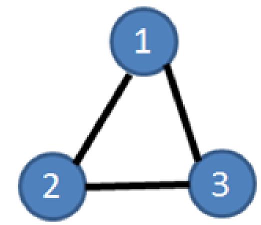


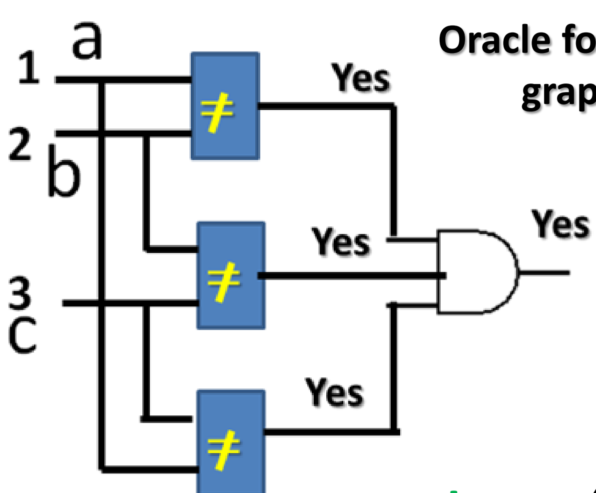
These two graphs need two colors



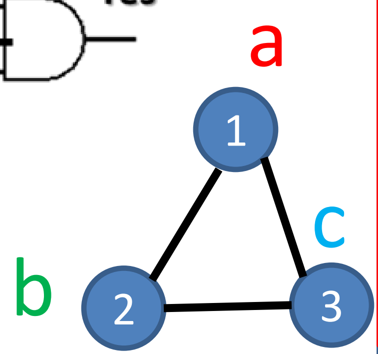


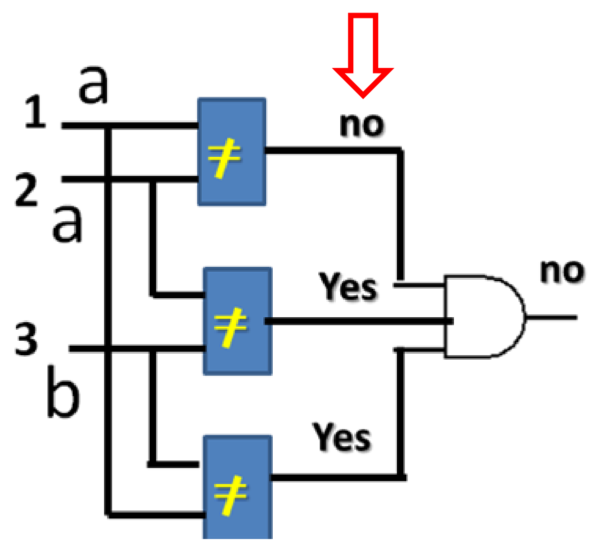
This graph G1 needs three colors





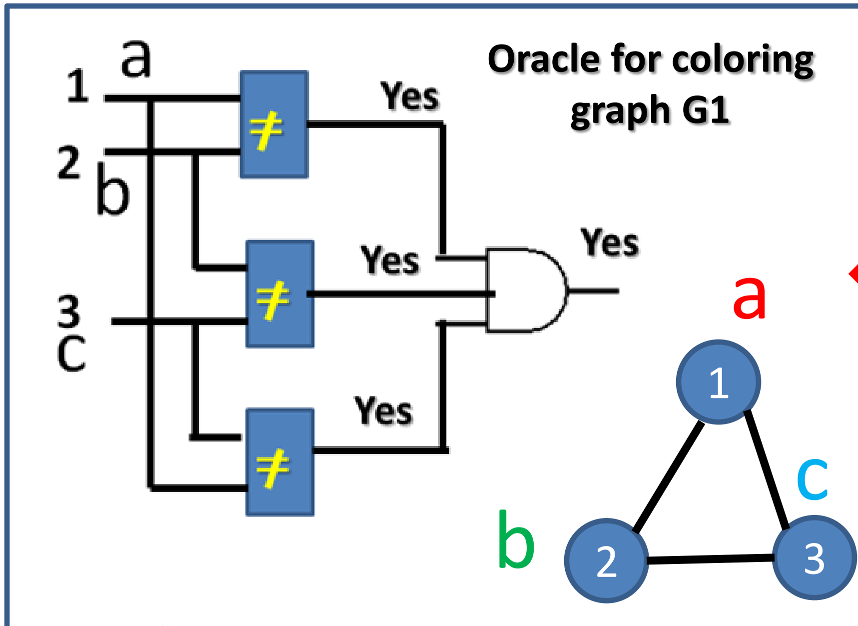
Oracle for coloring graph G1





Chromatic number of our graph is 3 because we cannot color it with 2 colors a and b

Example of CSP – Graph Coloring



Our Methodology – what we do with the oracle?

Our methodology creates such general oracle from the specification

Backtracking Simulator in PROLOG

Reversible Logic

Linear Programming

Standard Boolean Gates

Quantum Circuit

Standard Boolean Gates converted to Hamiltonians

FPGA

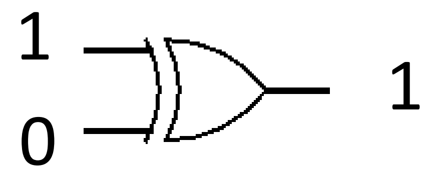
IBM quantum computer

Quantum Simulator

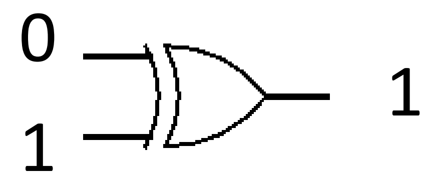
FPGA with Random Number Generators

We think about gates and blocks as characteristic functions or relations

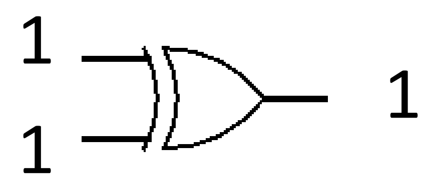
Think about a gate (a block, a Hamiltonian) as a little processor that likes to minimize its energy



I like it. I have minimum energy, I am happy, let us keep my state this way

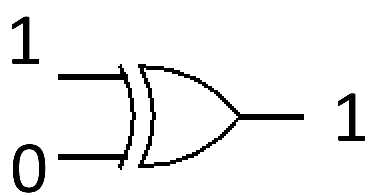


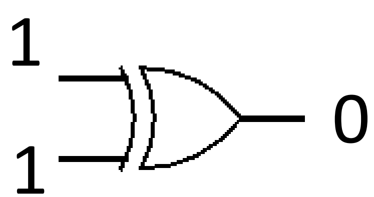
I like it. I have minimum energy, I am happy, let us keep my state this way



I hate it. I do not have minimum energy. I need to change signals.

I am happy again





Methodology to solve Constraint Satisfaction Problems and Optimization Problems based on Generalized Oracles

Key points of our methodology

1. Bottom-up rather than Top-Down
2. Gates and blocks, not Unitary or Permutative Matrices
3. Use libraries
4. Blocks: logic, arithmetic, predicates, problem-specific (like controlled counters)
5. Subset-of-set, versus permutative, versus mappings, versus combinations without repetitions, etc.
6. Encoding of data (short codes, one-hot, thermometer, etc)
7. Calculate abstract complexity such as number of iterations of oracle.
8. Calculate detailed complexity in terms of number of qubits, number of gates, number of pulses, etc.
9. Reusing of problem reductions, design tricks and verified blocks
10. Special synthesis algorithms for symmetric functions, ESOPs, PSOE, factorized forms, adders, controlled-adders, constant-multipliers, comparators, etc. = we use engineering design practices.
11. Using QSHARP, QISKIT, QUIPPER
12. Using Prolog for Oracle Simulation.
13. Ternary logic.

Grover Algorithm and Quantum Oracles

Second Example

SAT Oracle

This oracle is specific to the Satisfiability problem (SAT): **F = (a+b') (a'+c)**

Weight 16

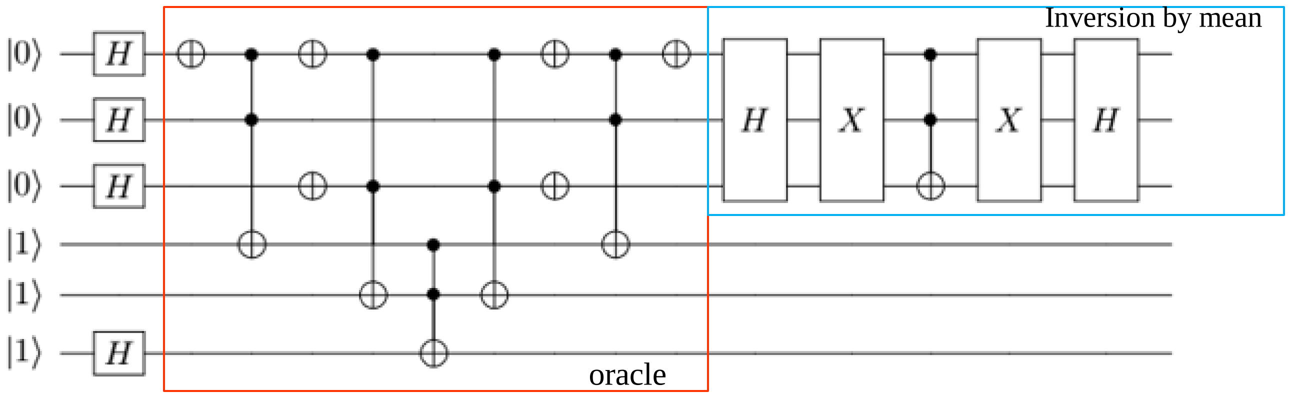
Weight 8

Weight 4

Weight 2

Weight 1

Remember weights



One repetition of GroverLoop for 2-SAT Problem, full circuit has two

Problem has 4 solutions, as shown in Truth Table 2.2.

This is designed to test the working of Grover's algorithm on more practical applications as many problems may be reduced to SAT based problems. The Toffoli oracle mentioned previously has been replaced with this oracle and the results will be slightly different from the previous one due to the presence of multiple satisfying conditions as seen in the truth table as shown below:

Numbers for solutions

2+1=3

4+2+1=7

16+4+2+1=23

16+8+4+2+1=31

Table 2.2 Truth table for F

	A	B	C	A+B'	A'+C	F= (A+B')(A'+C)
0	0	0	0	1	1	1
1	0	0	1	1	1	1
2	0	1	0	0	1	0
3	0	1	1	0	1	0
4	1	0	0	1	0	0
5	1	0	1	1	1	1
6	1	1	0	1	0	0
7	1	1	1	1	1	1

Based on search space, There needs $\frac{\pi}{4}\sqrt{8} \approx 2$ iterations to get the final results

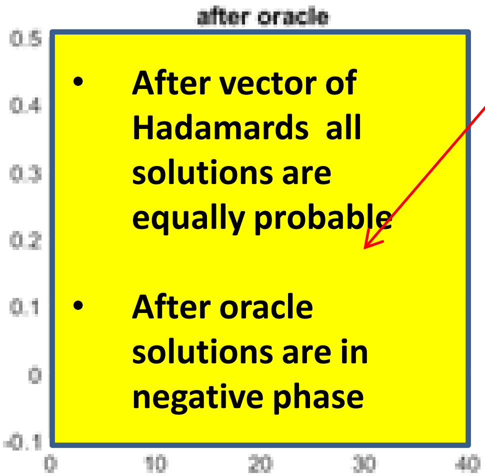
The results of simulation for each iteration are as follows:

This would be for 2 variables

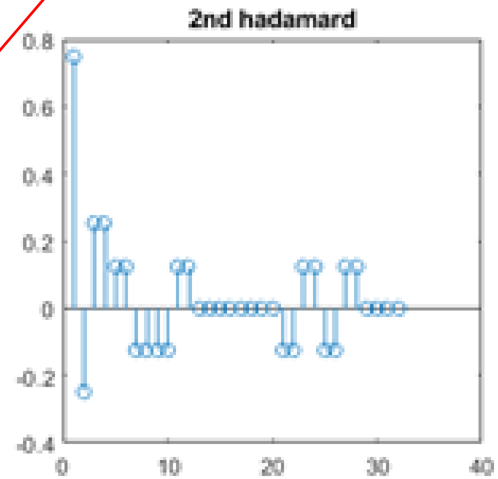
Superposition

RESULTS OF FIRST ITERATION

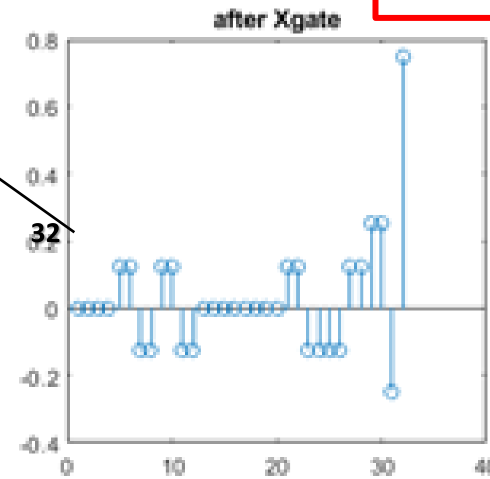
$$1/2|00\rangle+1/2|01\rangle+1/2|10\rangle+1/2|11\rangle$$

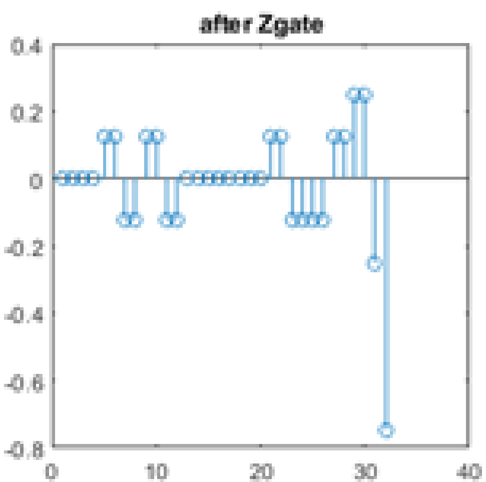


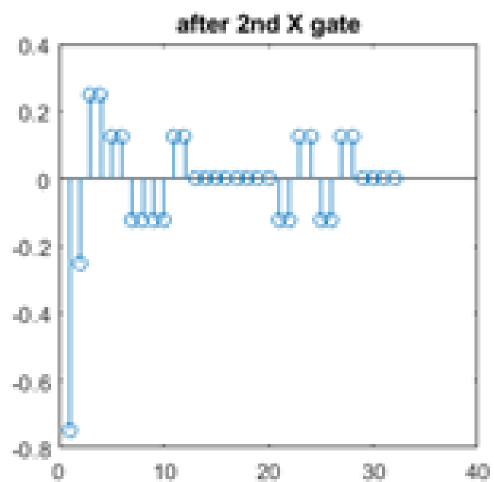
- **After vector of Hadamards all solutions are equally probable**
- **After oracle solutions are in negative phase**

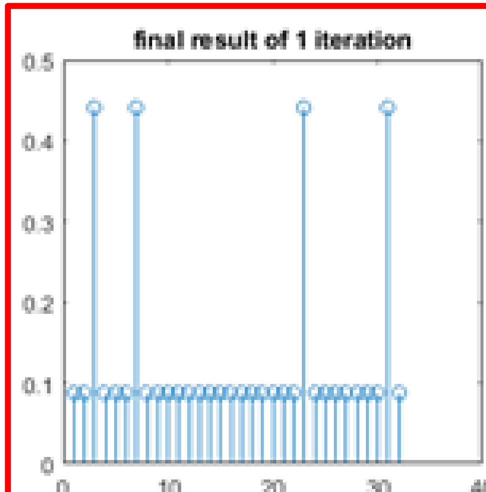


32



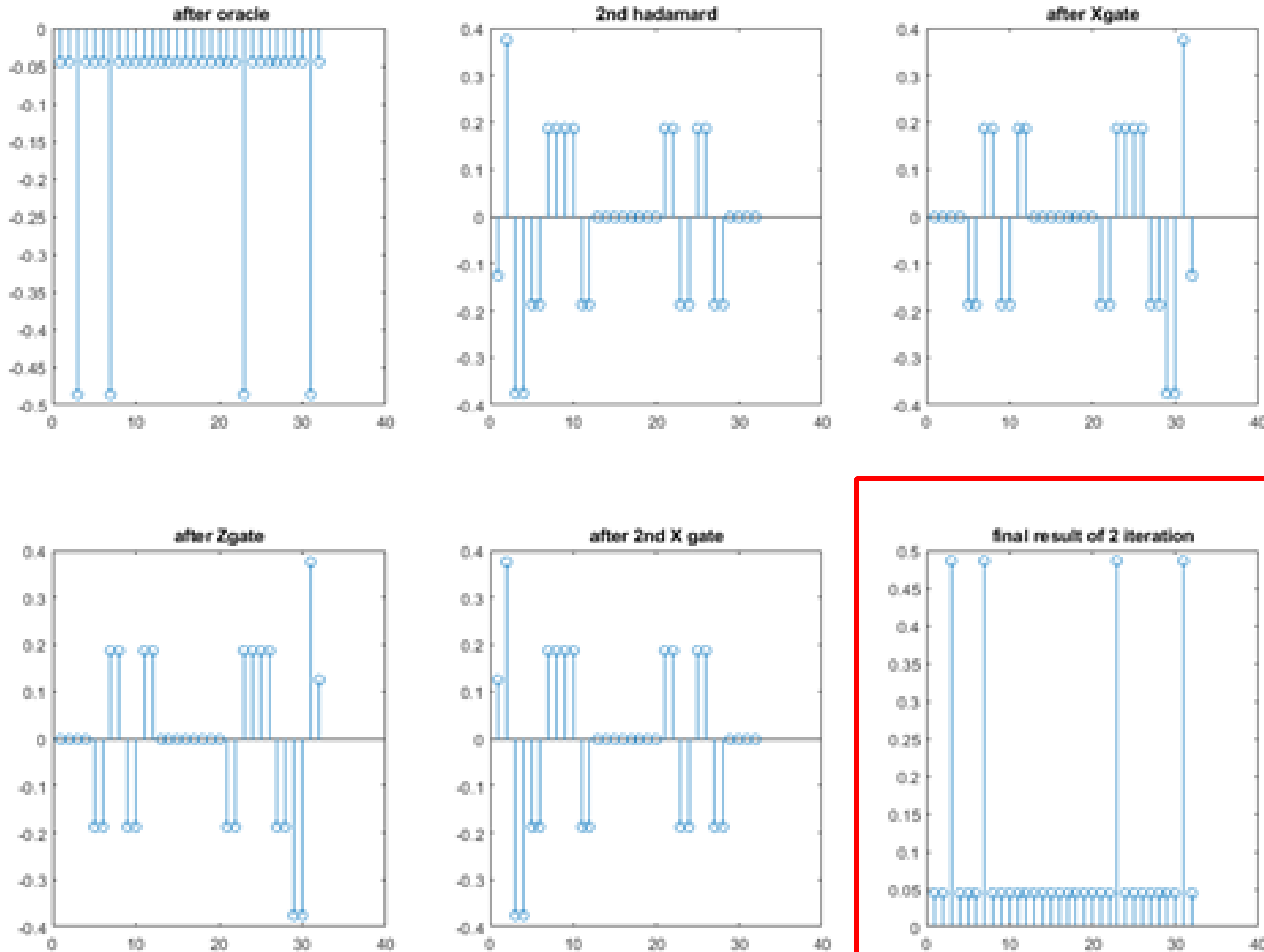






Magnitude of our 4 solutions is amplified

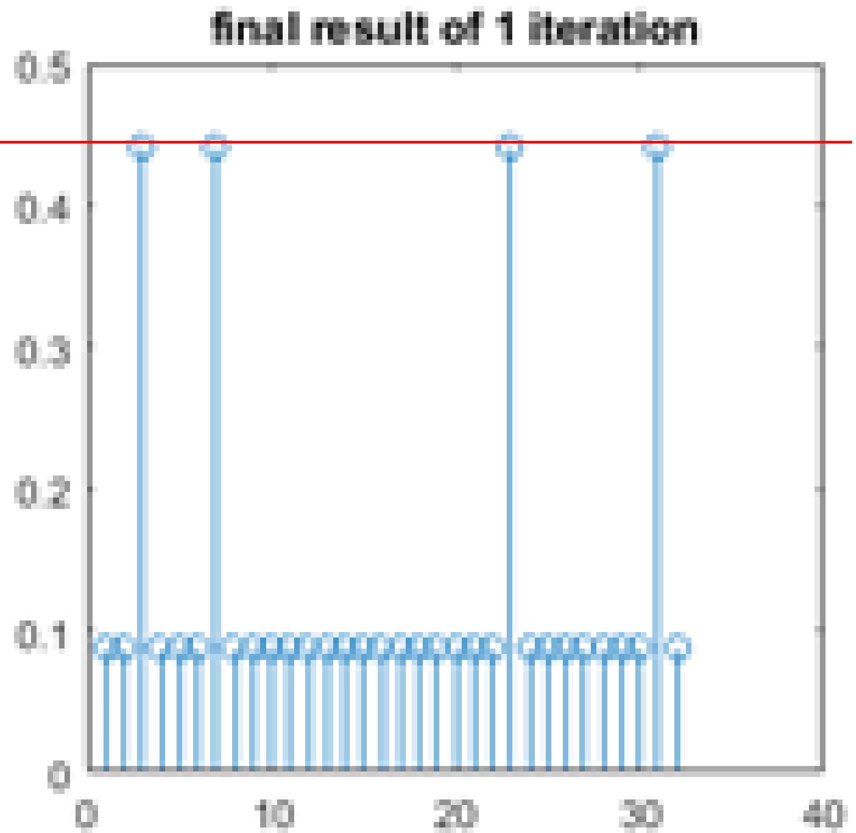
Phase of our 4 solutions is different

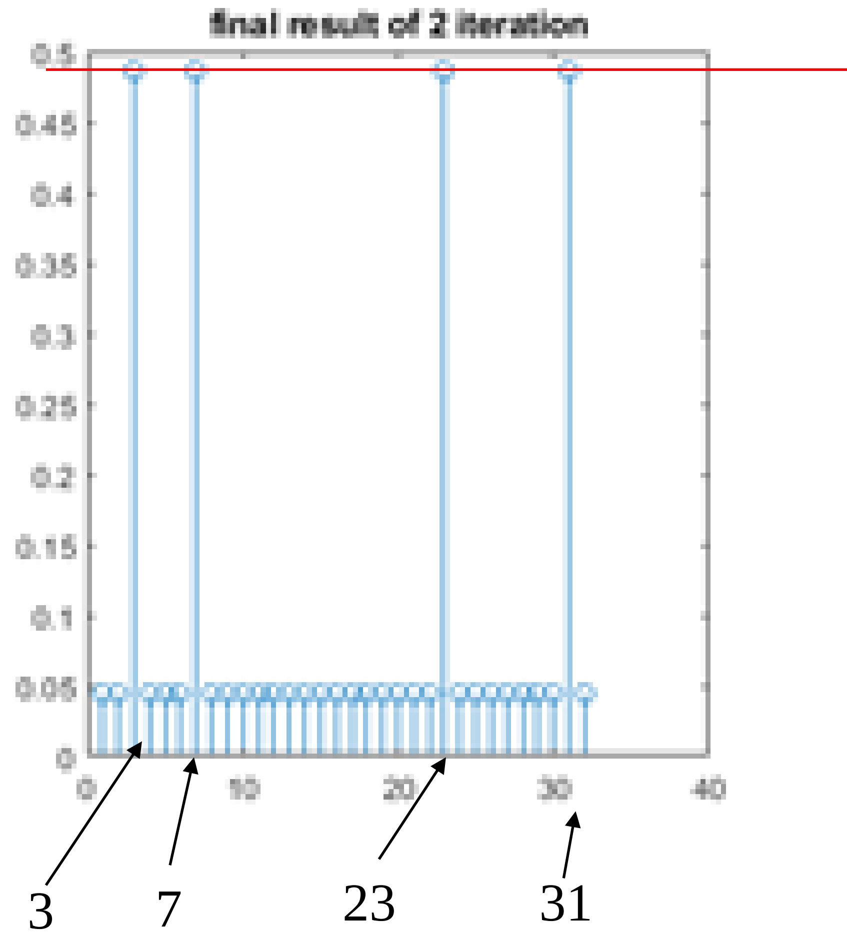


Magnitude of our 4 solutions is amplified

Result of 2nd iteration

Let us compare states after **first** and **second** iteration



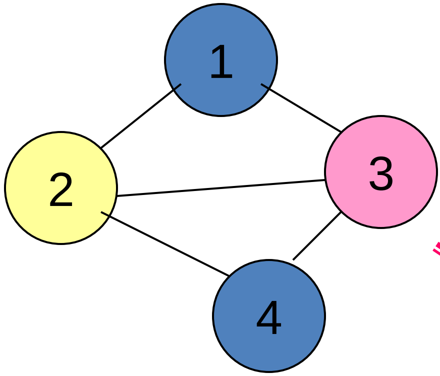


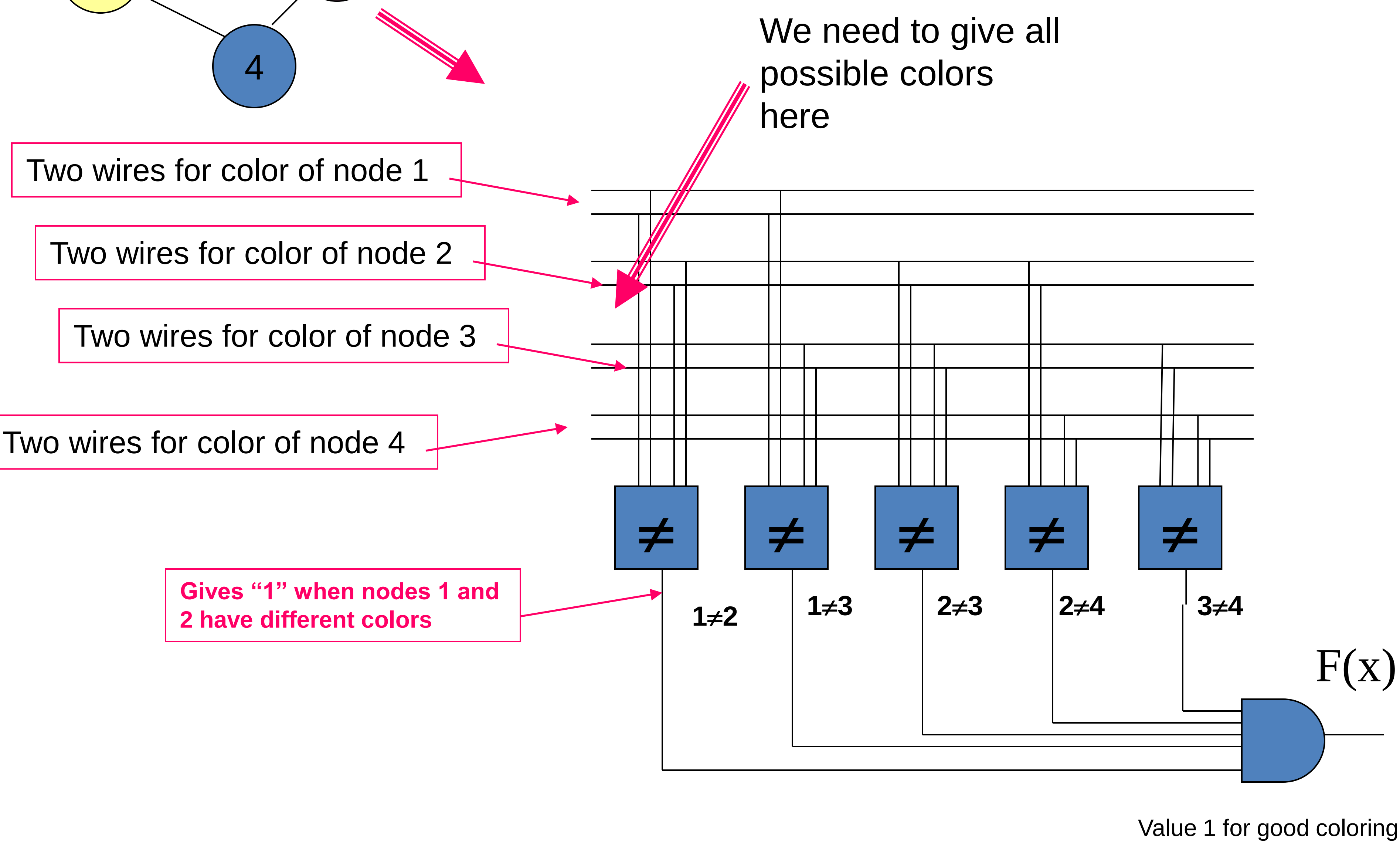
Even measuring now the probability of finding one of solutions is high

Measuring after two iterations the probability
measuring one of solutions is very high

Grover for Graph Coloring

A Simple Graph Coloring Problem



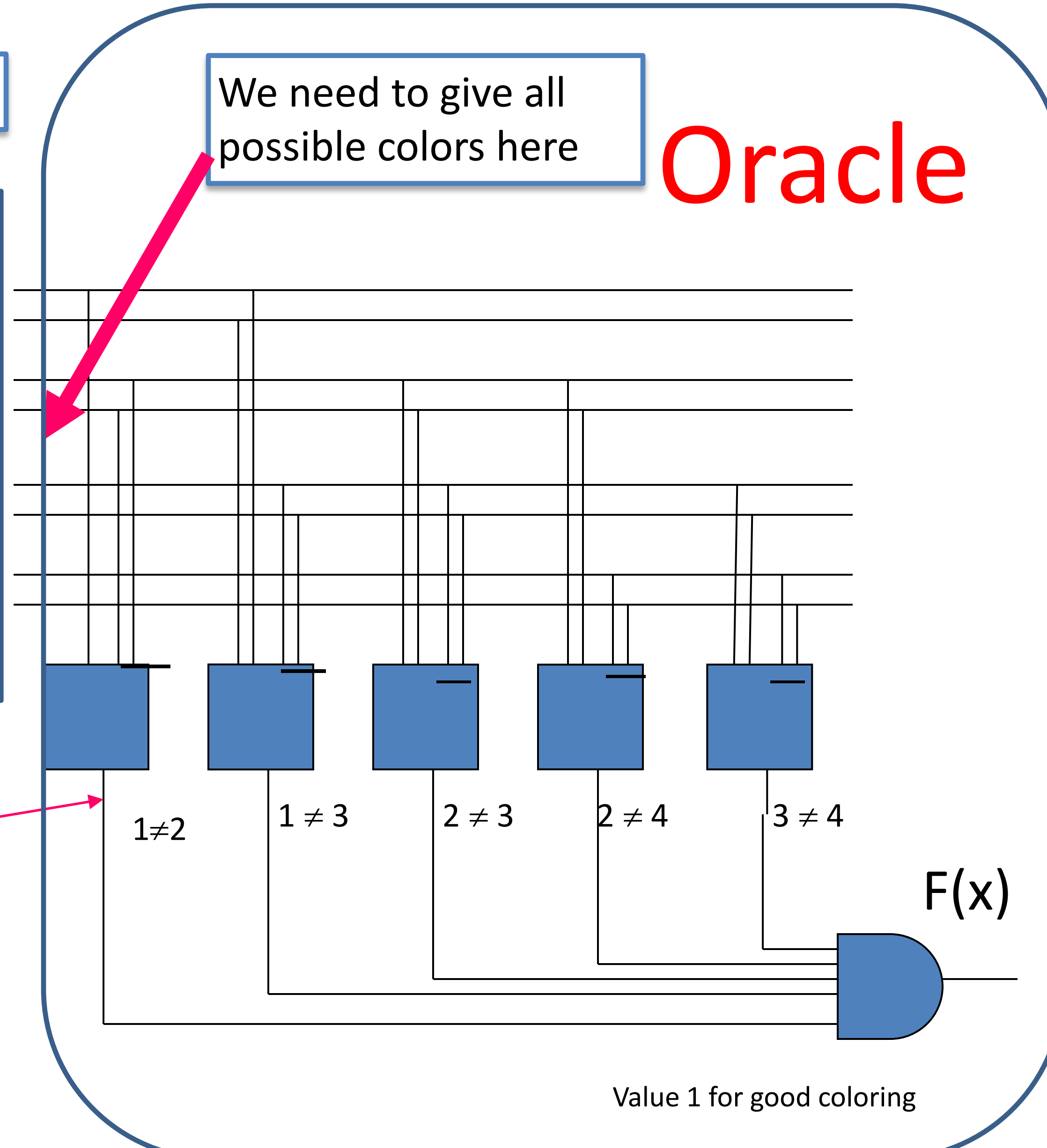


Sequential Generator and Oracle for Graph Coloring

Sequential Generator

Counter with 2^8 states

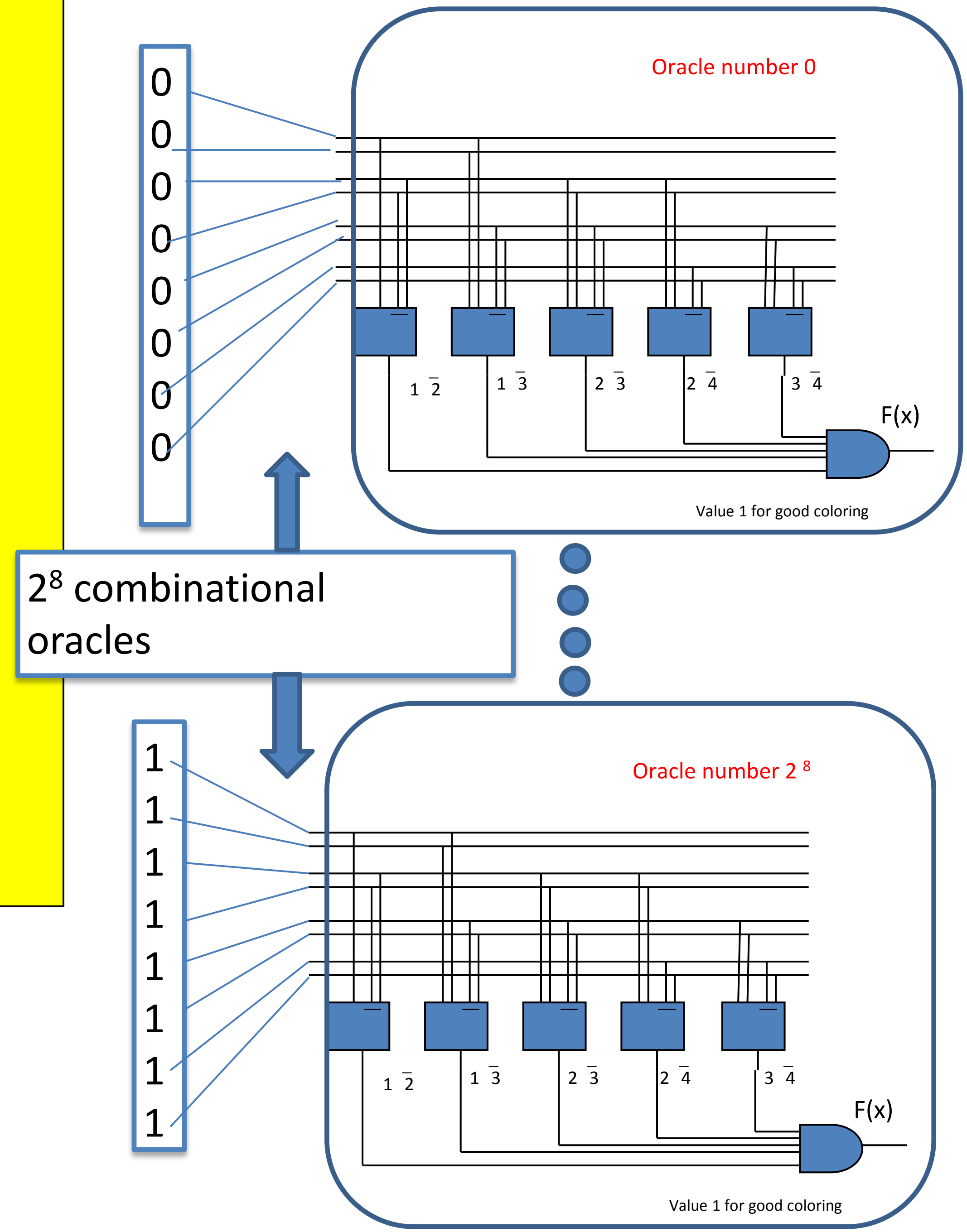
Gives “1” when nodes 1 and 2 have different colors



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

Hypothetical Parallel Computer

for Graph Coloring and 2^8 combinational oracles



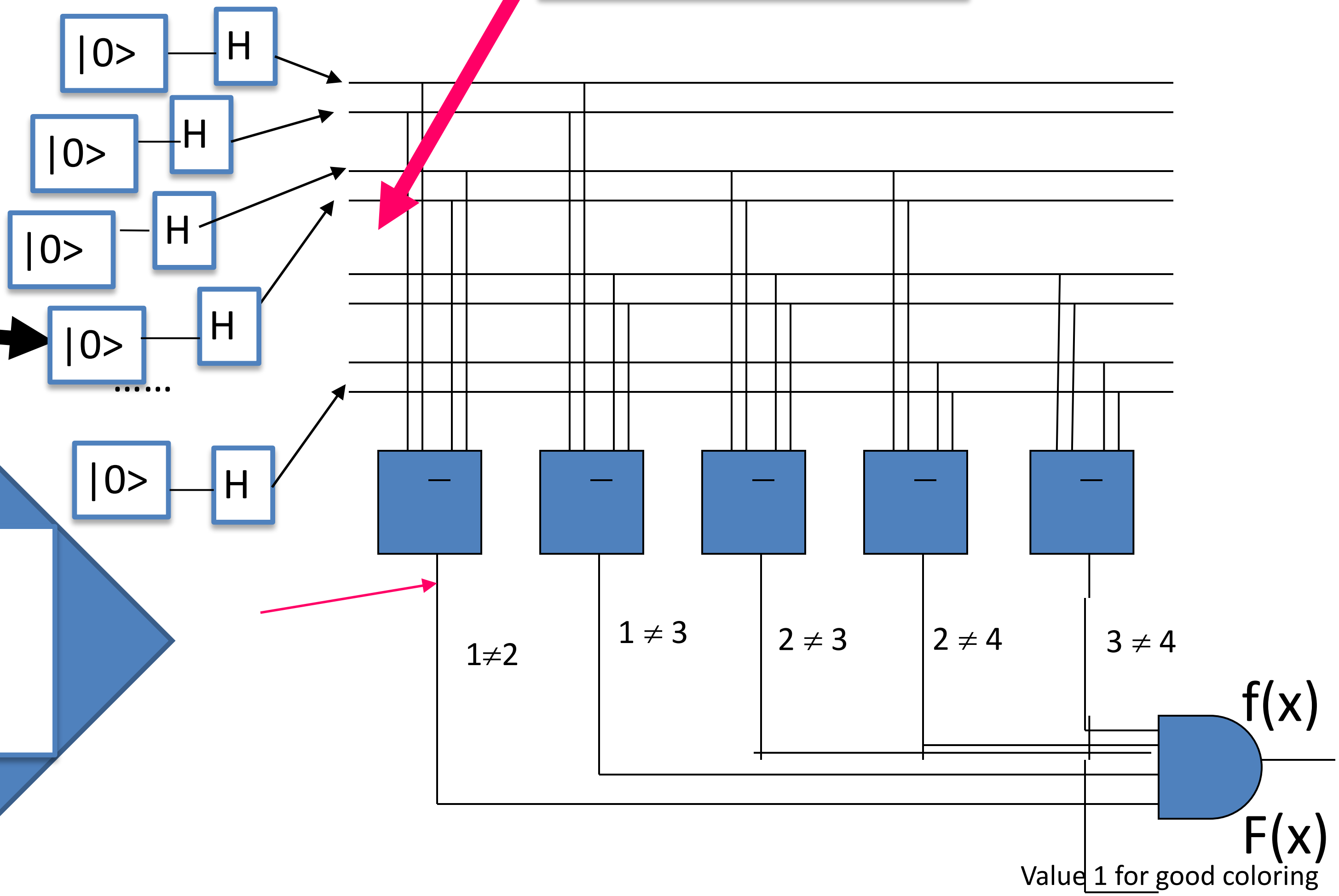
In case of quantum circuit, combinational parallelism is realized by superposition of states.

Circuit calculates on 2^n states <u>at the same time</u>

Simple 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



Now we have quantum oracle with quantum reversible gates

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

From Classical to Quantum Oracles

Classical Oracle

1. Non-reversible,
2. Built from standard gates like AND,OR,EXOR, NOT.
3. Calculate candidates sequentially

Quantum Oracle

1. Reversible,
2. Built from standard quantum gates like Toffoli, CNOT, NOT.
3. Calculate candidates in parallel, thanks to quantum superposition

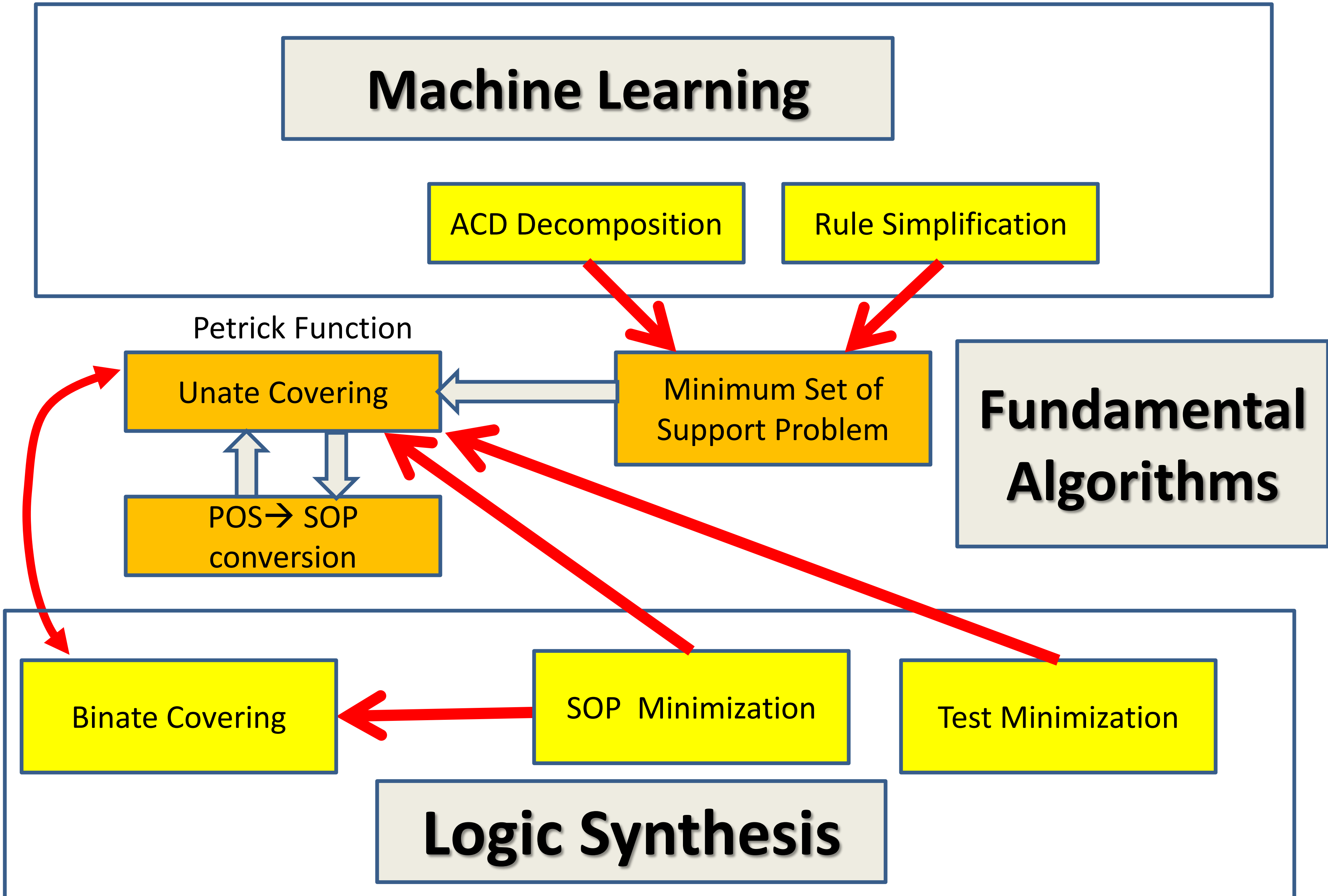
Reducing Optimization Problems to Decision Problems with Modified Oracles

Grover for Minimum Set of Support Problem

IDEA

Hybrid Classical-Quantum Computer creates a sequence of modified oracles for Grover

Example: Minimum Set of Support



Example: Minimum Set of Support

ab\cd	00	01	11	10
00	x	x	1	x
01	1	x	x	1
11	x	x	0	x
10	0	x	x	0

Karnaugh Map of an incomplete function of 4 variables a,b,c,d. (Mathematically a relation)

Compare evey true minterm with every false minterm

From Relational Specification to Set of simple Boolean Equations

OFF\ON	0011	0100	0110
1111	$a+b$	$a+c+d$	$a+d$
1000	$a+c+d$	$a+b$	$a+b+c$
1010	$a+d$	$a+b+c$	$a+b$

From Individual Equations to a single equation

OFF\ON	0011	0100	0110
1111	$a+b$	$a+c+d$	$a+d$
1000	$a+c+d$	$a+b$	$a+b+c$
1010	$a+d$	$a+b+c$	$a+b$

Local Equations of Boole

Single Equation of Boole

CLASSICAL

QUANTUM

$$(a+b).(a+c+d).(a+d).(a+c+d).(a+b).(a+b+c).(a+d).(a+b+c).(a+b)$$

Using Laws of Boolean Algebra

$$(a+b).(a+d)$$

POS

Oracle

solution

$$a + bd$$

SOP

Encoding and explanation of Grover Oracle for this problem

(a+b).(a+d) → **Oracle** → *a + bd*

POS   SOP

ENCODING
a = 1000
b = 0100
c = 0010
d = 0001
ab = 1100
acd = 1011

(a+b)

(a+d)

ab\cd	00	01	11	10
00	0	0	0	0
01	0	1	1	0
11	1	1	1	1
10	1	1	1	1

This cell 1000 is encoding of solution a

Solution bd

ab\cd	00	01	11	10
00	0	0	0	0
01	0			0
11				
10				

Solution a

Finding solution **a** we want to exclude all included in **a**, such as **ab**, **ac**, **abc**, **abcd**

Idea of modifying oracles

Grover Oracle for all solutions to this problem

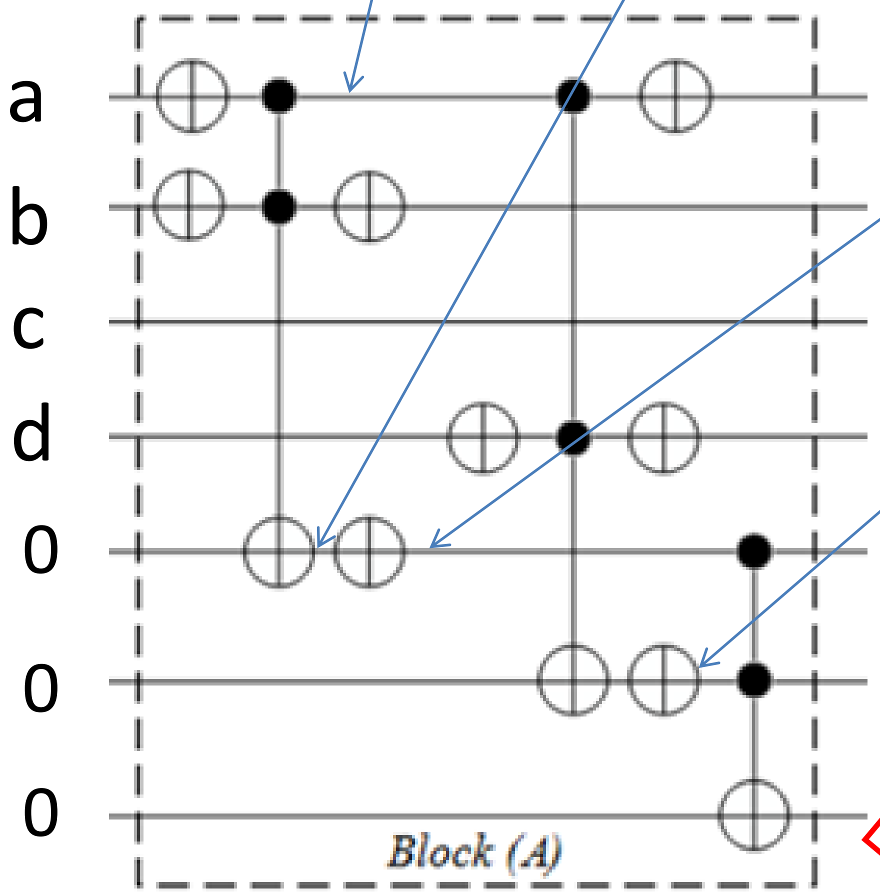
Use De Morgan Rule

$\bar{a}$

$\bar{a}\,\bar{b} \oplus 0$

(a+b)

(a+d)



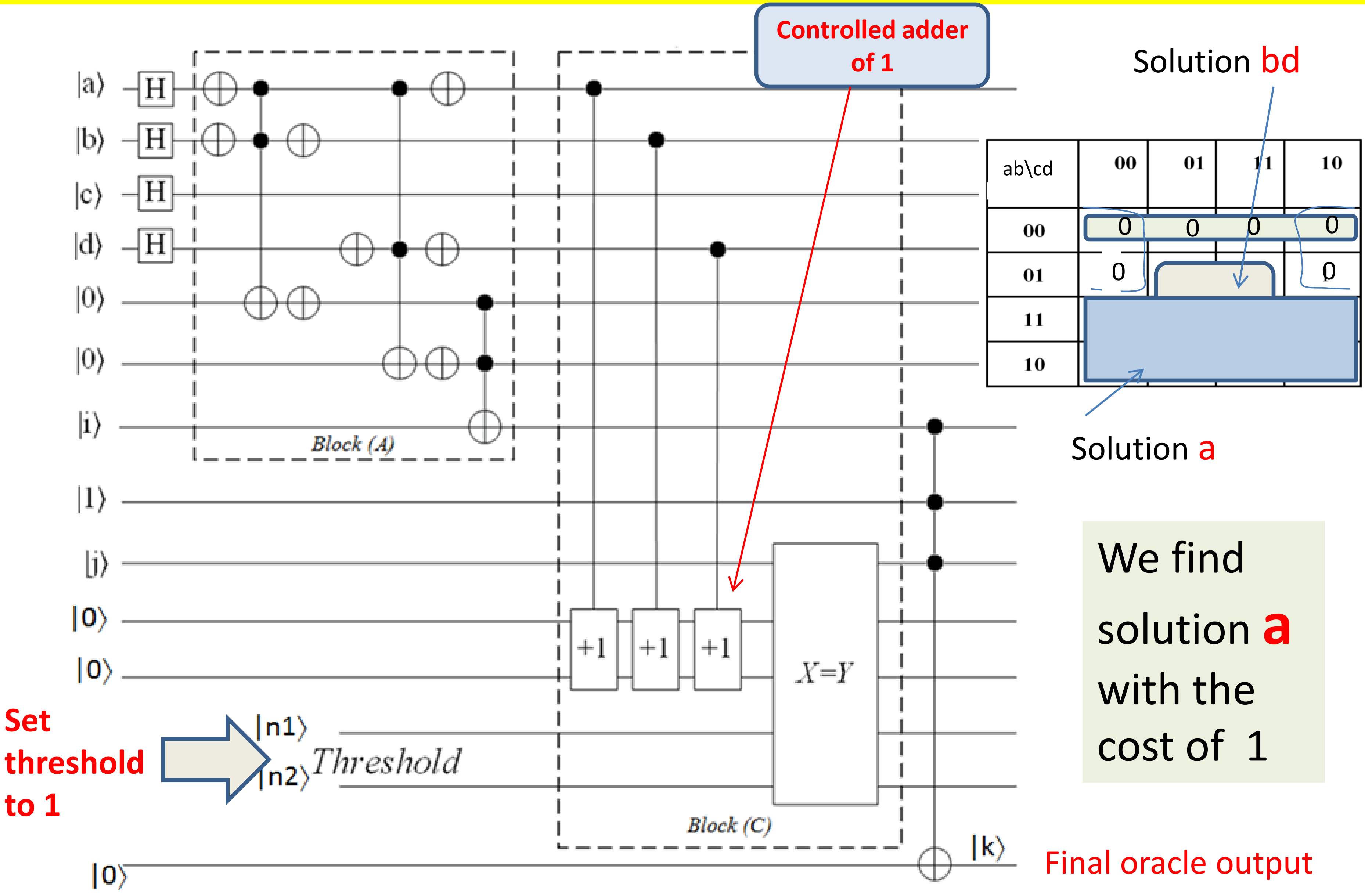
ENCODING

a = 1000
b = 0100
c = 0010
d = 0001
ab = 1100
acd = 1011

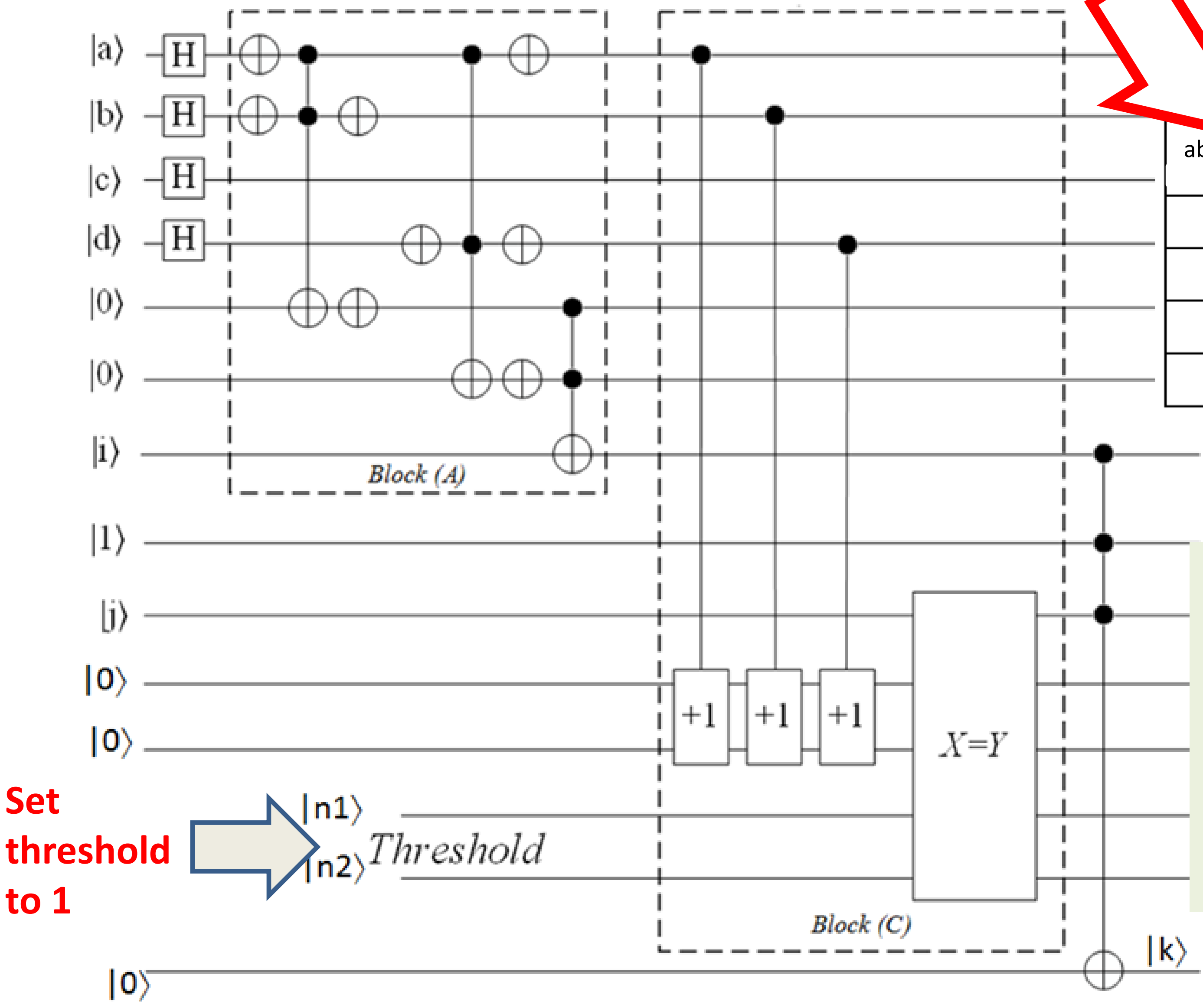
$(a+b).(a+d) \oplus 0$
$= (a+b).(a+d)$

This oracle would find all solutions to problem but most of them would be non-minimal

First Grover Oracle for this problem



First Grover Oracle for this problem



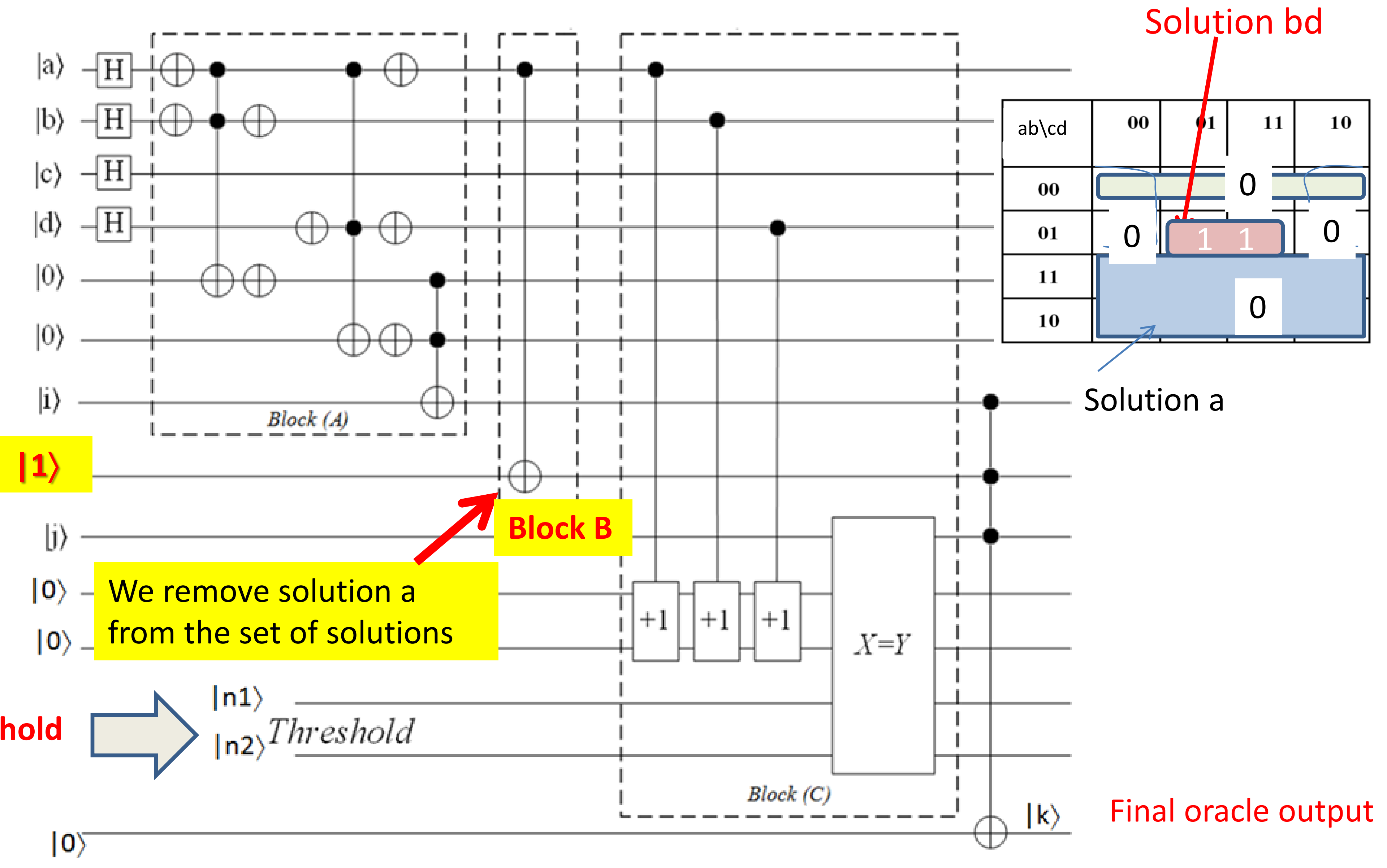
Solution bd

ab\cd	00	01	11	10
00	0	0	0	0
01	0			0
11	0	0	0	0
10	0	0	0	0

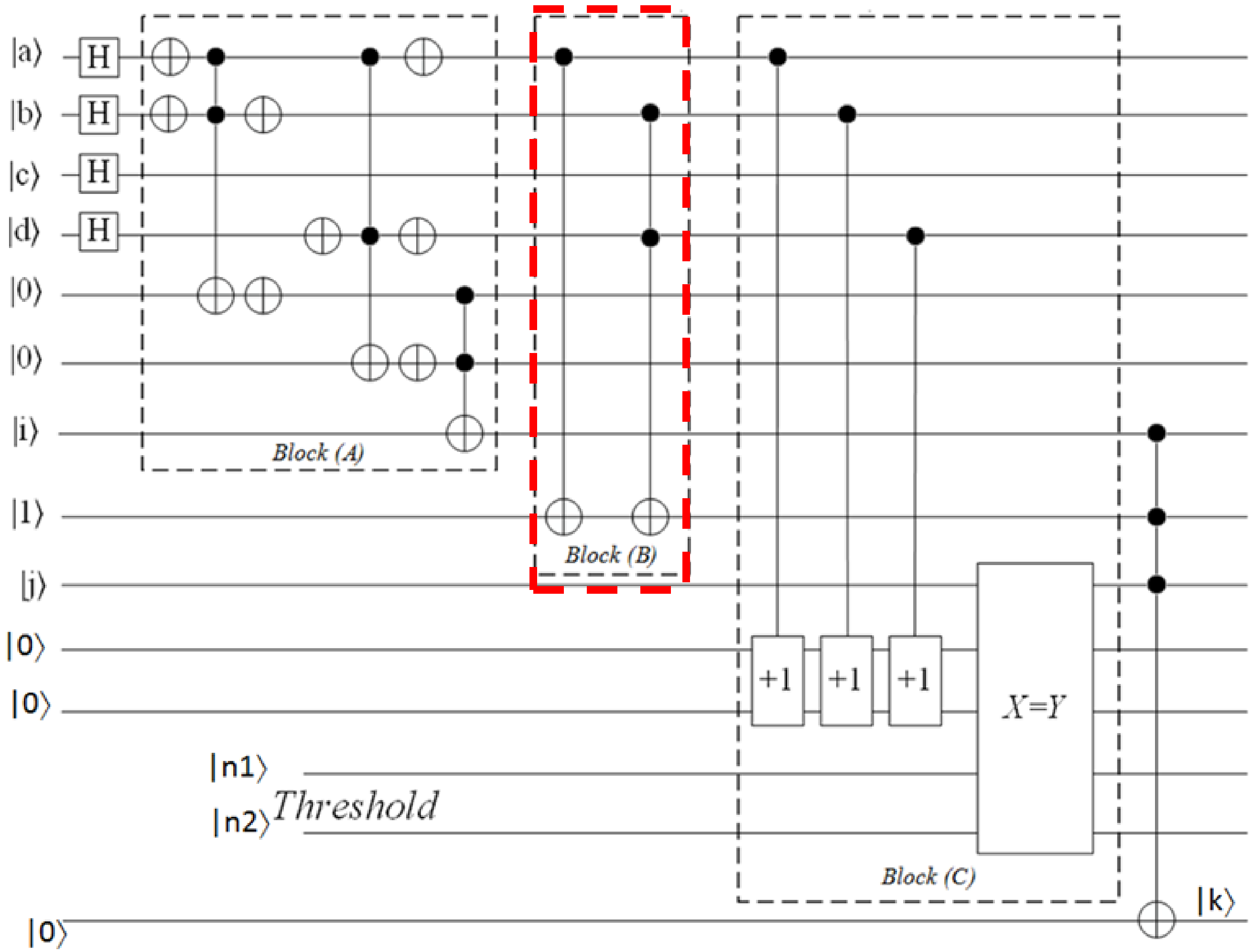
Solution a

- We find solution a with the cost of 1
- We remove all solutions included in a, such as ab, ac, abc, etc

Second Grover Oracle for this problem



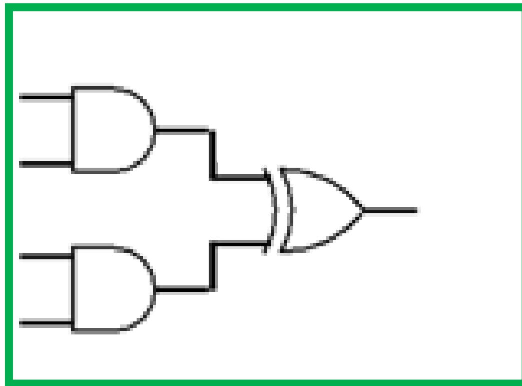
Third Grover Oracle for this problem



ab\cd	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	0	0	0	0
10	0	0	0	0

- **There is no solution**
- **Grover gives something random, every time different.**
- **We can verify or we can run earlier Quantum Counting to count number of solutions**

Invertible Quantum



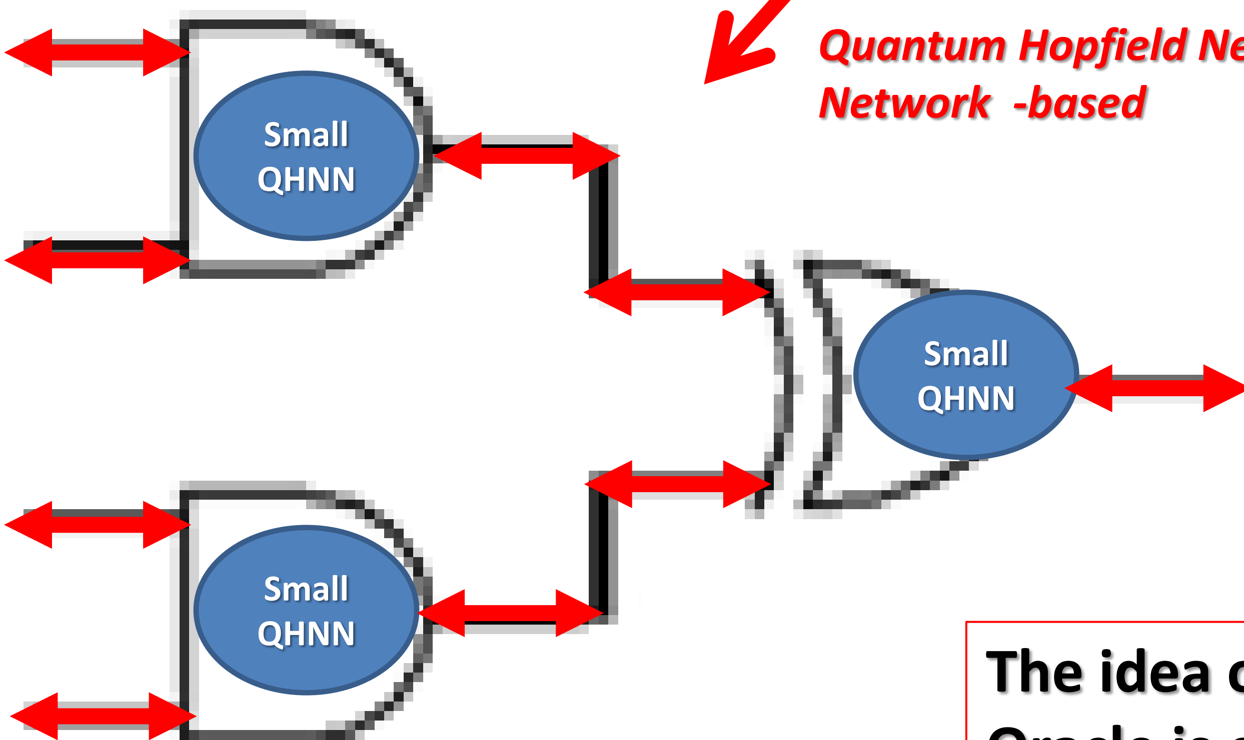
Quantum Grover

Classical FPGA

Invertible FPGA

Spin and other nano-technologies

Quantum Hopfield Neural Network -based



The idea of Generalized Oracle is central to all these approaches

Conclusions

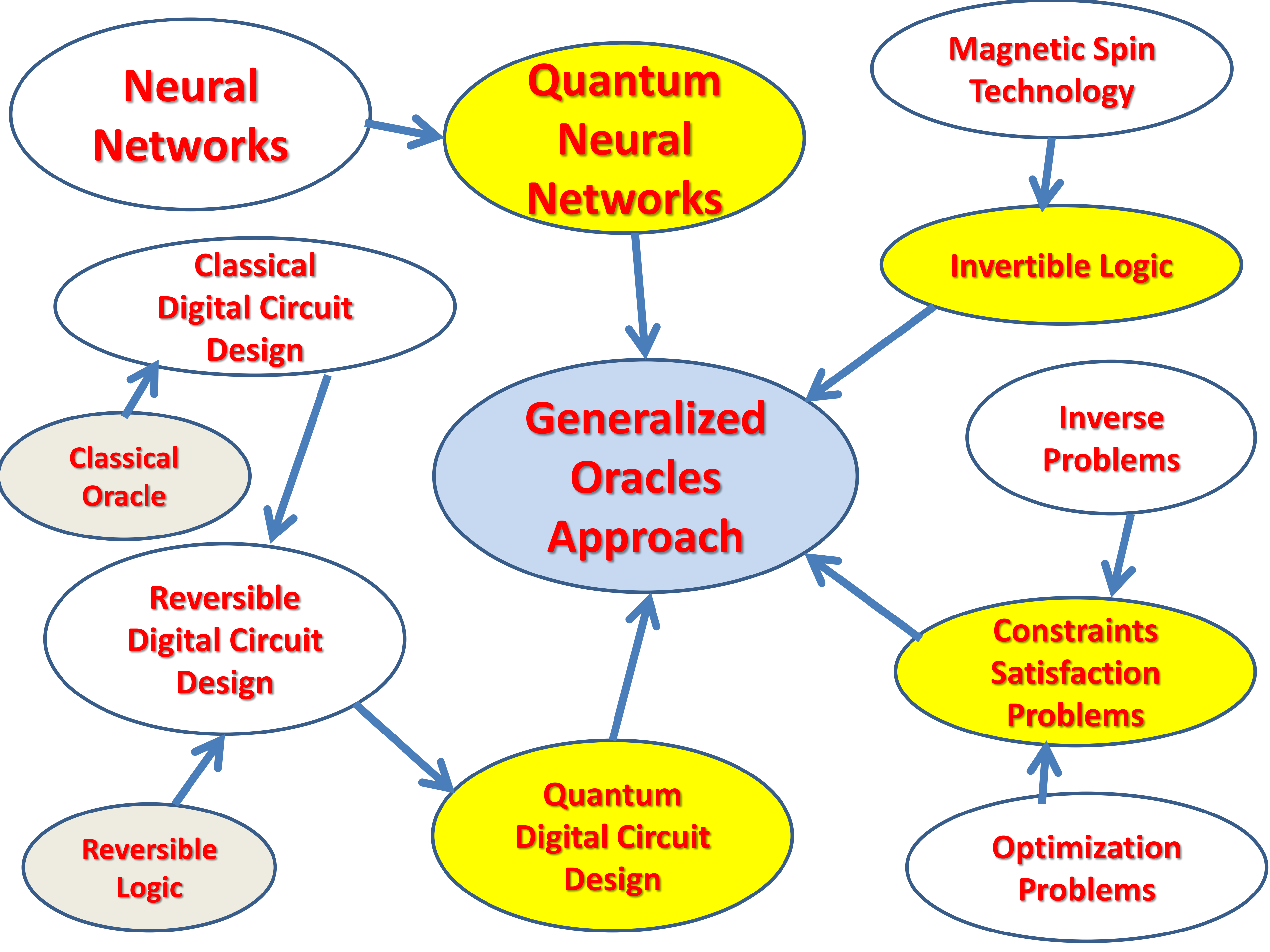
- Presented approach allows to solve **all** *Constraint Satisfaction Problems*.
- **Optimization problems** are solved by sequence of CSP with modified oracles.
- Oracles can be built in several technologies:
  1. **FPGA**,
  2. magnetic spins
  3. **quantum**.
- Quantum oracles use *reversible logic*.
- Invertible oracles use *invertible logic*.
- In FPGA we can model both classical and invertible logic oracles.

Conclusions (cont)

- When FPGA is used for invertible logic the most important is to create a <u>very high quality</u> Random Number Generator.
  - These are subjects of our research
    1. Commercial Quantum RNG based on Hadamard Gates
    2. LFSR, EXOR forest, jitter generators
    3. FPGAs with analog-like random number generators.

- **FPGA** realizations
- Prolog **Simulations**
- Quantum realizations of Grover Methodology
- Quantum realizations of **Invertible** Logic with **QHNN** (Quantum Hopfield Neural Net)

OUR RESEARCH


Neural Networks
Quantum Neural Networks
Magnetic Spin Technology
Invertible Logic
Classical Digital Circuit Design
Generalized Oracles Approach
Classical Oracle
Inverse Problems
Reversible Digital Circuit Design
Constraints Satisfaction Problems
Quantum Digital Circuit Design
Reversible Logic
Optimization Problems

Questions often asked

- How practical is this?
- When we will have large quantum computers?
- Error Detection and Correction.
- Can we **combine the powers of** Invertible Logic and Reversible Logic in Quantum technology to create superior algorithms?

Current research

- What is the methodology? What is the math?
  1. Combinatorics, coding theory, numeric representations
  2. Algorithms and Complexity
  3. Spectral Methods
  4. Digital Design
  5. Algebra, graph theory, partition theory, cube calculus theory

No time to discuss

- We do not discuss:
  - Quantum Technologies
  - or Magnetic Spin Technologies

- **I welcome any question!**