HyQSAT: A Hybrid Approach for 3-SAT Problems by Integrating Quantum Annealer with CDCL

Siwei Tan , Mingqian Yu , Andre Python , Yongheng Shang ,
Tingting Li , Liqiang Lu*, and Jianwei Yin*

Reporter： Siwei Tan Supervisor ： Jianwei Yin , Liqiang Lu

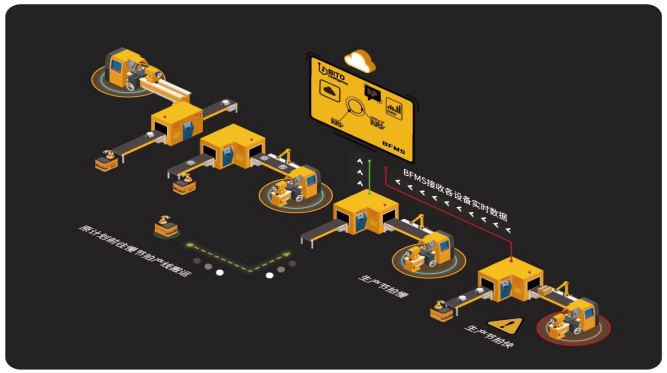
College of Computer Science and Technology
,Zhejiang University

The Application of SAT Problem

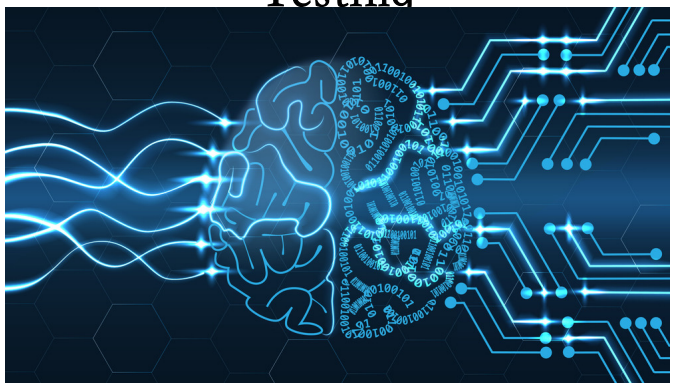
Propositional satisfiability problem (SAT)

Cryptography

Software Testing



Planning



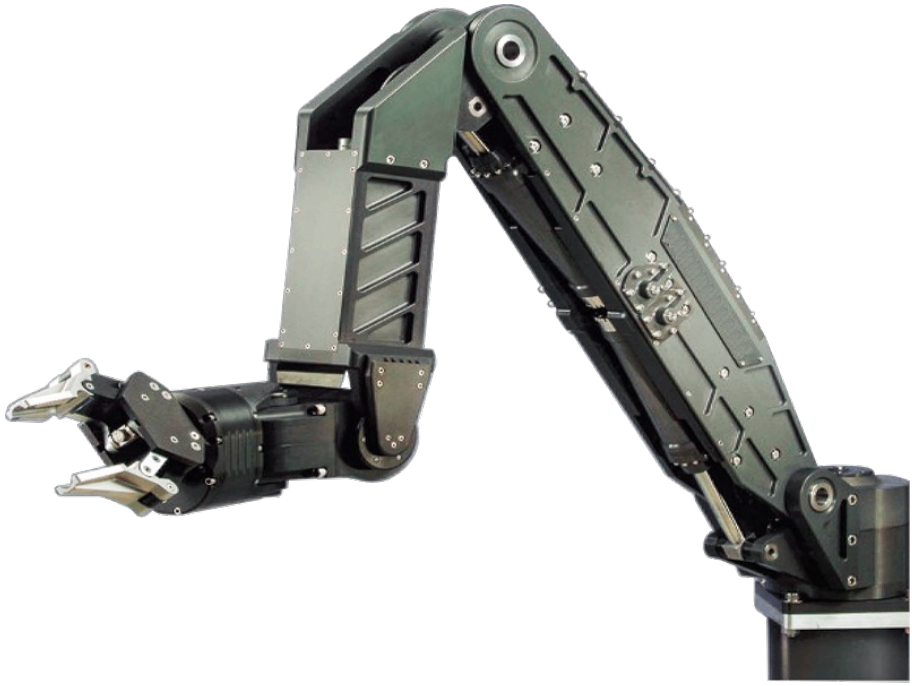
Artificial Intelligence

Protein structure analysis
Knowledge inference ……

Example: A Motor Vehicle Parts Production Line

A product line can produce 1,000 products per day.

20,000 products need to be produced, including A, B, C, D….



Constraints:

A and B must be produced together;

B must be produced together with one of E, F or G;

C cannot be produced with E together ;

……

The optimal classical algorithm takes 3 days to find the optimal schedule.

The Formulation the SAT Problem: An Example

A SAT problem C in a **conjunctive normal form** with variables x_1 , x_2, x_2, x_4 :

$$C = c_1 \wedge c_2$$

$$c_1 = x_1 \vee x_2 \vee x_3,$$

$$c_2 = \neg x_2 \vee \neg x_3 \vee x_4$$

A **clause** means : x_1 **or** x_2 **or** x_2

A **solution** of the given problem:

$$x_1 = x_2 = 0$$
$$x_3 = x_4 = 1$$

All clauses need to be satisfied.

3-SAT problem: each clause has no more then 3 variables. **The first NP-complete problem。**

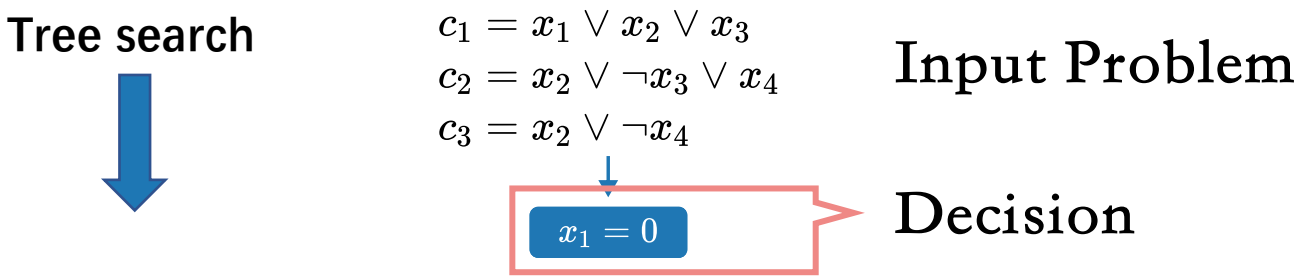
The Optimal Classical Algorithm: CDCL Algorithm

Tree search

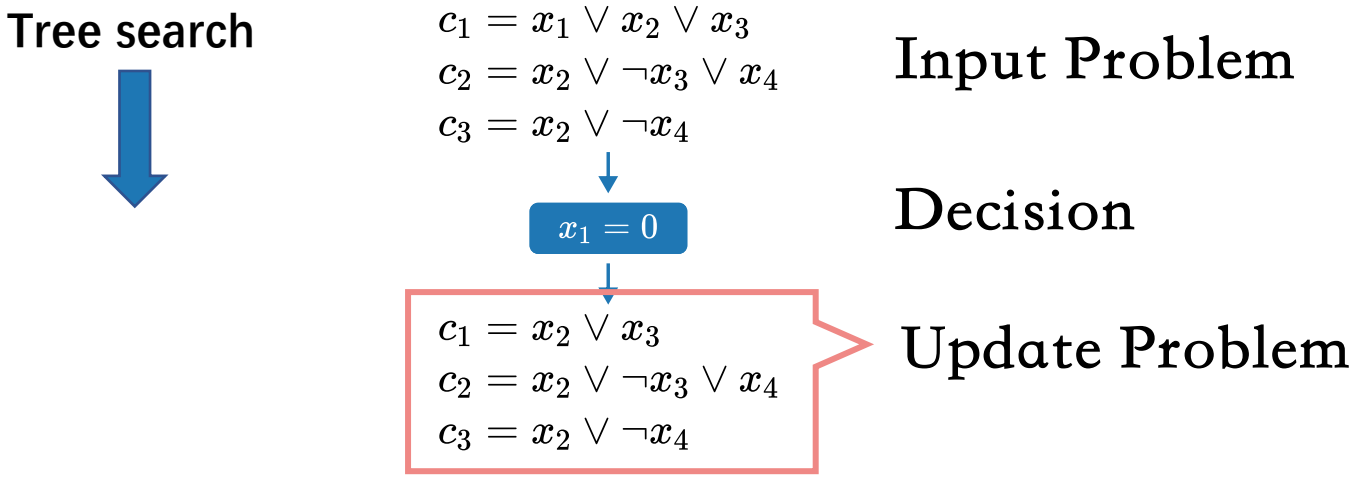
$c_1 = x_1 \vee x_2 \vee x_3$
$c_2 = x_2 \vee \neg x_3 \vee x_4$
$c_3 = x_2 \vee \neg x_4$

Input Problem

The Optimal Classical Algorithm: CDCL Algorithm



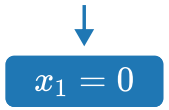
The Optimal Classical Algorithm: CDCL Algorithm



The Optimal Classical Algorithm: CDCL Algorithm

Tree search

$c_1 = x_1 \vee x_2 \vee x_3$
$c_2 = x_2 \vee \neg x_3 \vee x_4$
$c_3 = x_2 \vee \neg x_4$



$x_1 = 0$

$c_1 = x_2 \vee x_3$
$c_2 = x_2 \vee \neg x_3 \vee x_4$
$c_3 = x_2 \vee \neg x_4$

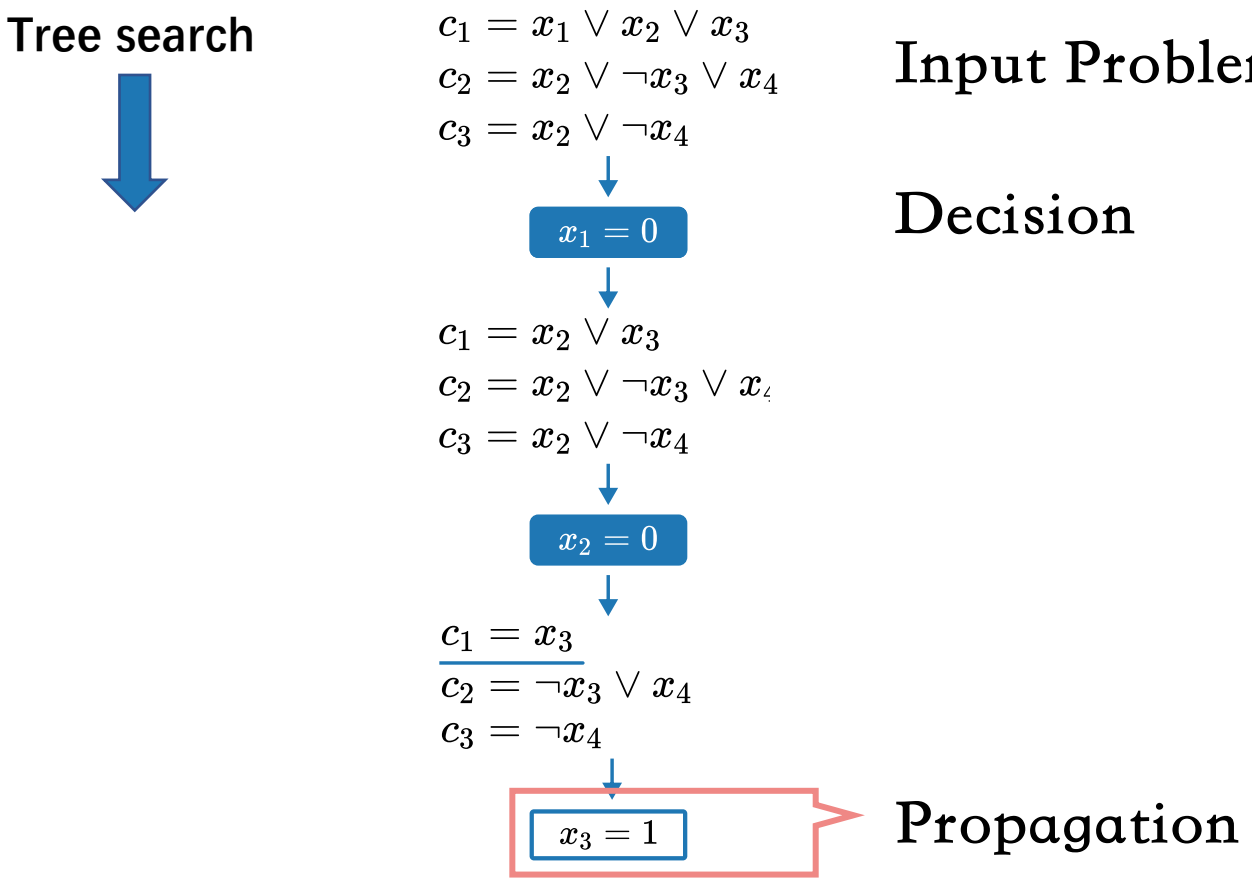
$x_2 = 0$

$c_1 = x_3$
$c_2 = \neg x_3 \vee x_4$
$c_3 = \neg x_4$

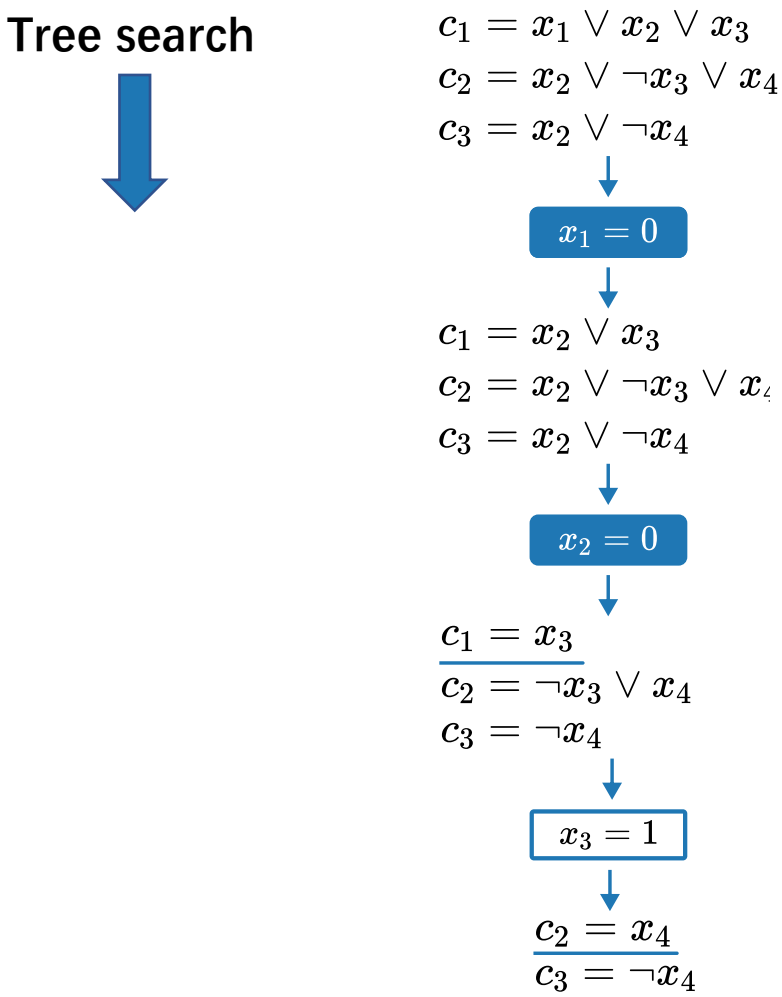
Input Problem

Decision

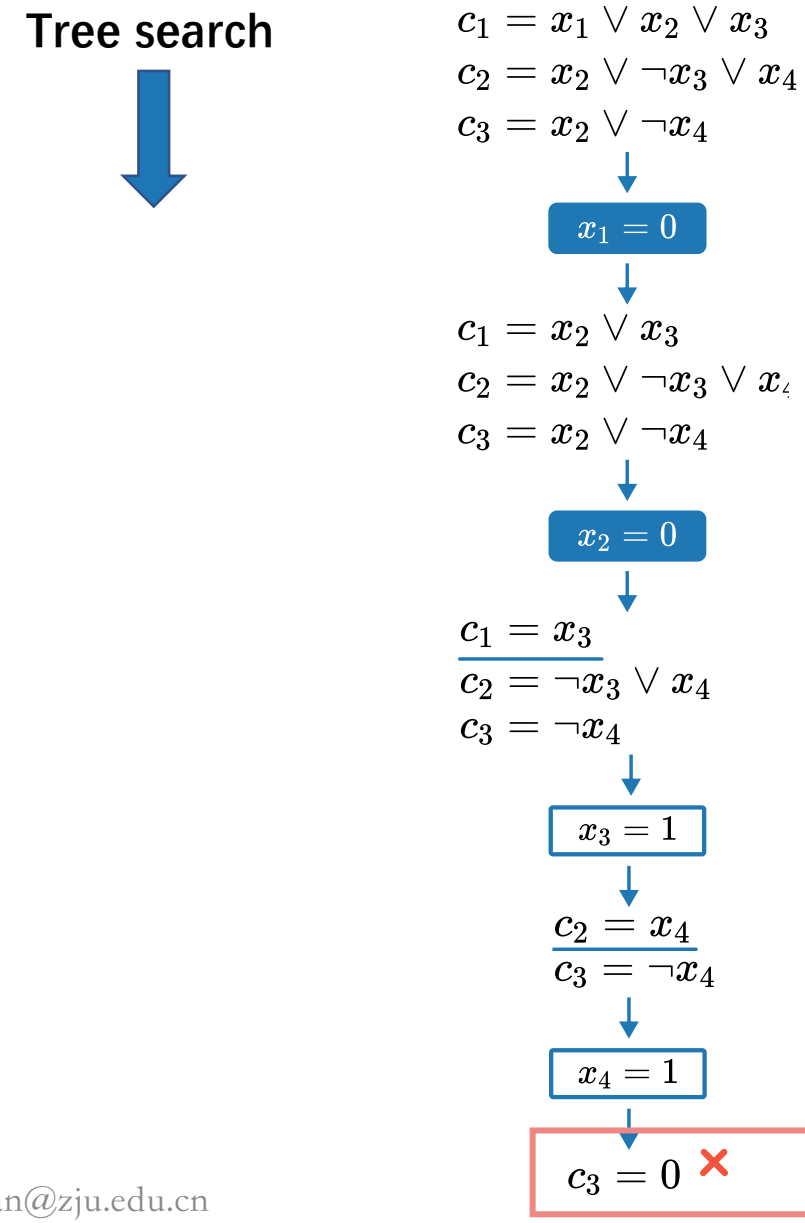
The Optimal Classical Algorithm: CDCL Algorithm



The Optimal Classical Algorithm: CDCL Algorithm

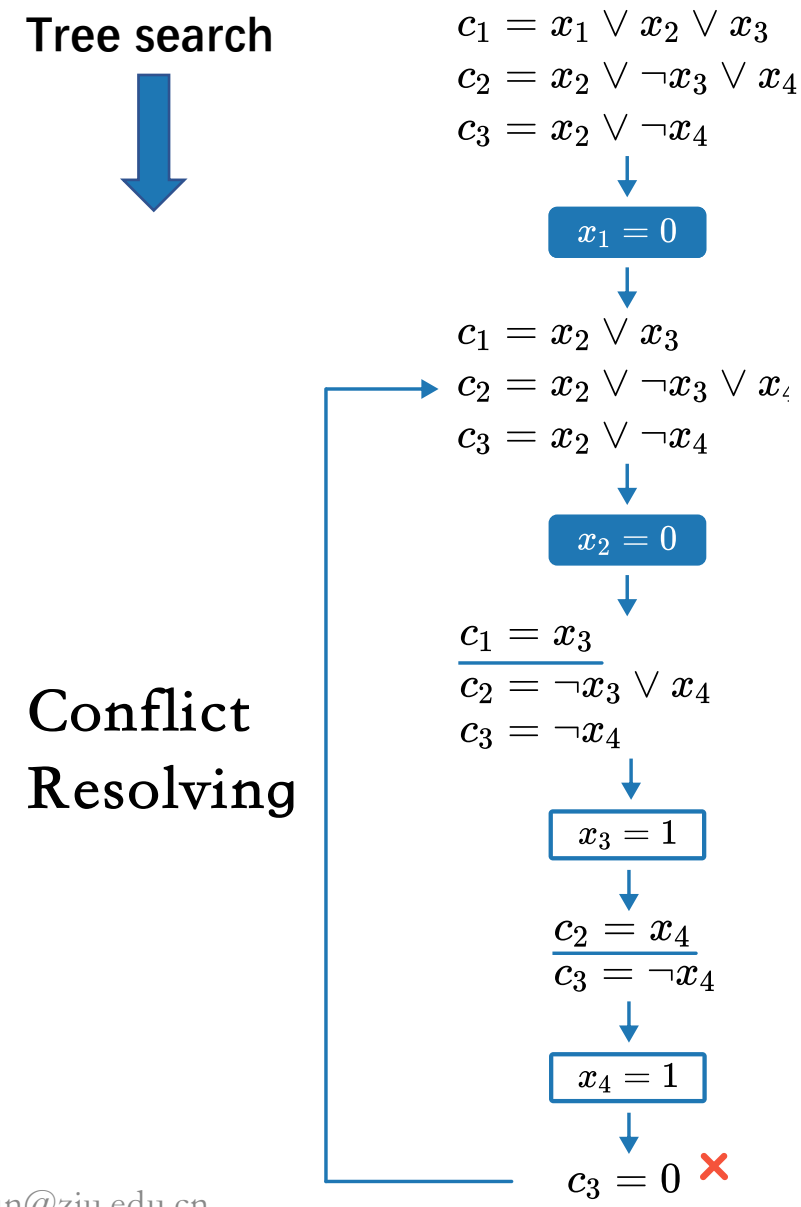


The Optimal Classical Algorithm: CDCL Algorithm



ZHEJIANG UNIVERSITY

The Optimal Classical Algorithm: CDCL Algorithm



Input Problem

Decision

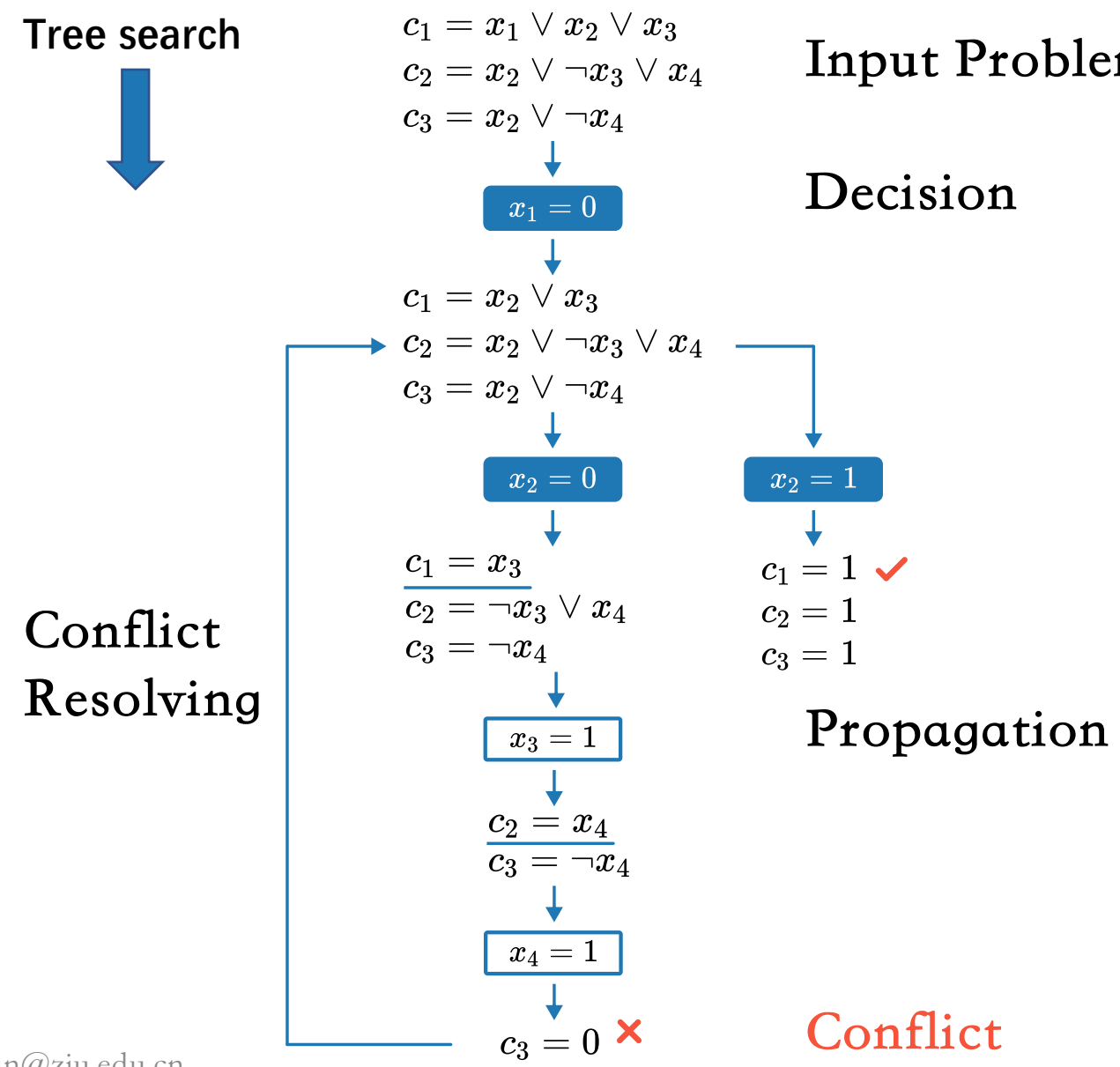
Conflict Resolving

Propagation

Conflict

ZHEJIANG UNIVERSITY

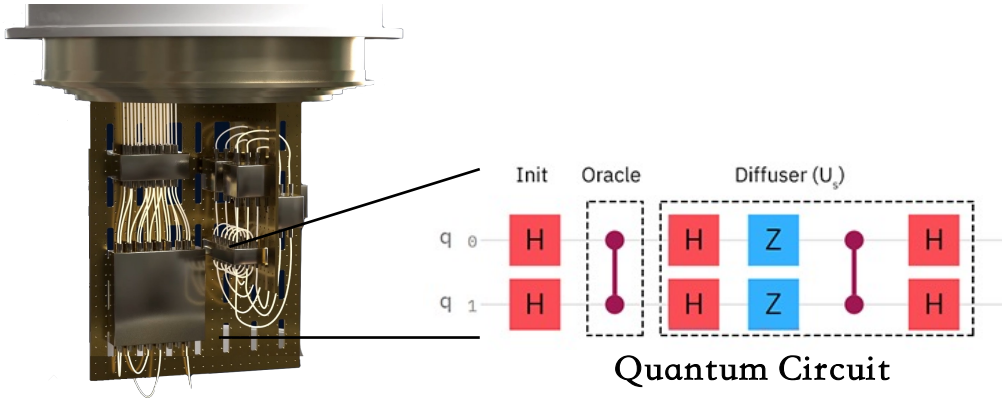
The Optimal Classical Algorithm: CDCL Algorithm



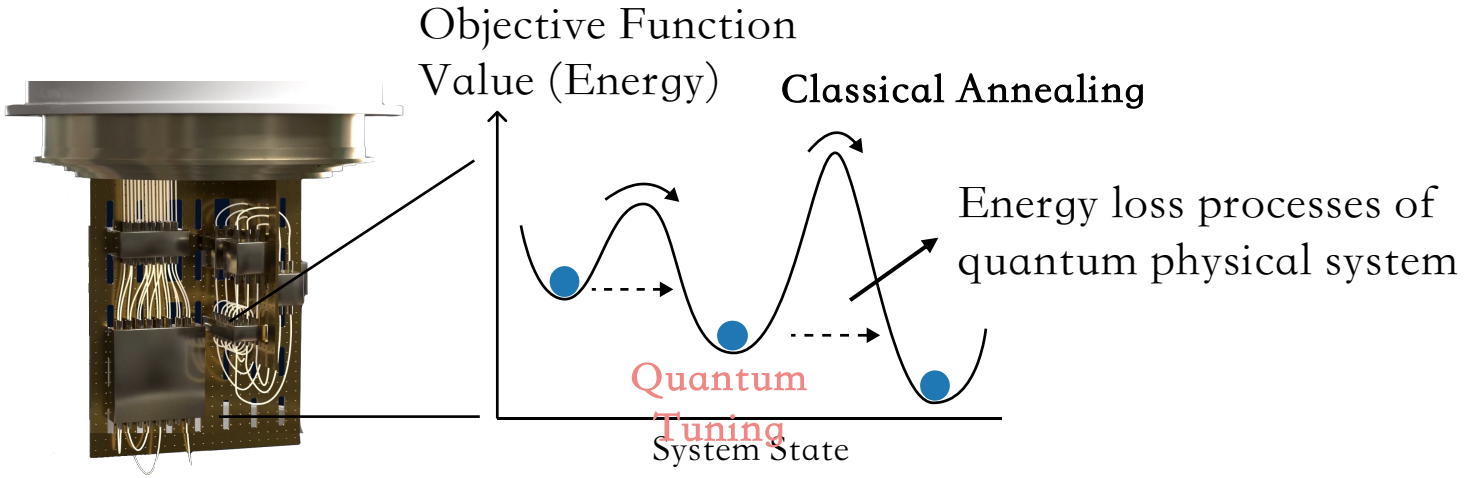
ZHEJIANG UNIVERSITY

Solving 3-SAT Problems by Quantum Computing

Gate-based Quantum Computer (Grover Algorithm、VQE Algorithm)

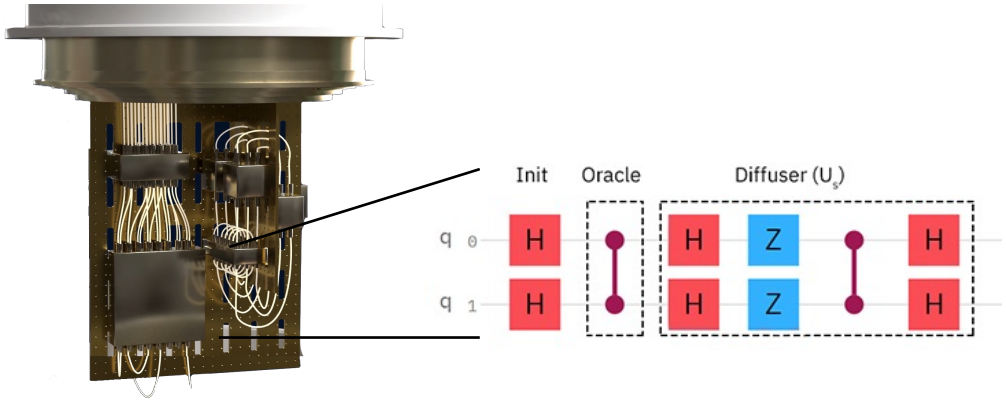


Quantum Annealer

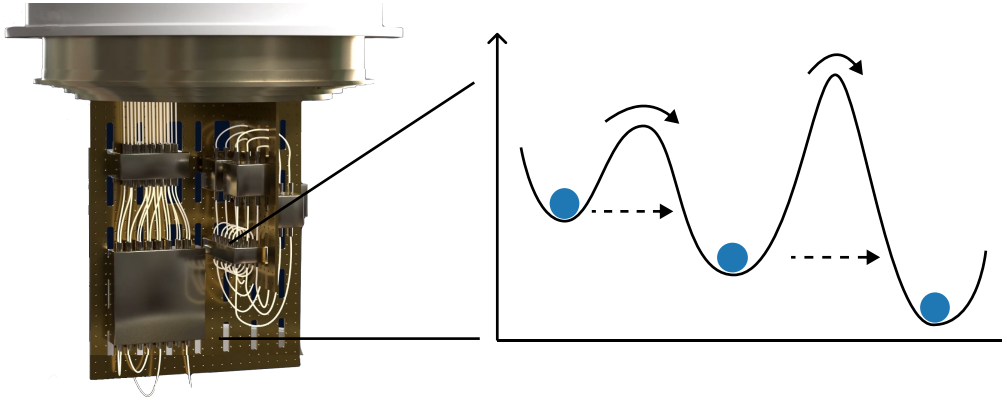


Solving 3-SAT Problems by Quantum Computing

Gate-based Quantum Computer (Grover Algorithm、VQE Algorithm)



Quantum Annealer



Quantum		Classical
Gate-based	Quantum Annealing (QA)	CDCL
Digital	Simulated	Digital
Quantum superposition	Quantum tunneling	Classical physics
$O(\sqrt{L})$	$O(e^{\sqrt{L}})$	$O(e^{L})$
~100 qubits	~2000 qubits	$>2^{30}$ bits
~10 variables	~50 variables	~1000 variables

Deploying 3-SAT problem to Quantum Annealer

The 3-SAT problem first should be transferred into the **minimization problem of a quadratic polynomial objective function**, formulated as:

Configured

$$\arg\min_{X} H_C(X) = I + \sum_{i=1}^{L} B_i x_i + \sum_{i=1}^{L} \sum_{j=i+1}^{L} J_{i,j} x_i x_j,$$

e.g.

$$C = c_1 \wedge c_2$$
$$c_1 = x_1 \vee x_2 \vee x_3,$$
$$c_2 = \neg x_2 \vee \neg x_3 \vee x_4$$

$$\begin{aligned} H_C(X, A) = 3 &+ x_1 + 3x_2 - 2x_3 \\ &- x_4 - 2a_2 + x_1 x_2 \\ &- x_2 x_3 - 2a_1 x_1 - 2a_1 x_2 \\ &+ a_1 x_3 - 2a_2 x_2 + 2a_2 x_3 \\ &+ a_2 x_4 \end{aligned}$$

Deploying 3-SAT Problem to Quantum Annealer

1

$$C = c_1 \wedge c_2$$
$$c_1 = x_1 \vee x_2 \vee x_3,$$
$$c_2 = \neg x_2 \vee \neg x_3 \vee x_4$$

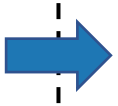
3-SAT Problem

Deploying 3-SAT Problem to Quantum Annealer

1

$$C = c_1 \wedge c_2$$
$$c_1 = x_1 \vee x_2 \vee x_3,$$
$$c_2 = \neg x_2 \vee \neg x_3 \vee x_4$$

3-SAT Problem



2

$$\begin{aligned} H_C(X, A) = 3 &+ x_1 + 3x_2 - 2x_3 \\ &- x_4 - 2a_2 + x_1x_2 \\ &- x_2x_3 - 2a_1x_1 - 2a_1x_2 \\ &+ a_1x_3 - 2a_2x_2 + 2a_2x_3 \\ &+ a_2x_4 \end{aligned}$$

Objective Function

Deploying 3-SAT Problem to Quantum Annealer

1

$$C = c_1 \wedge c_2$$
$$c_1 = x_1 \vee x_2 \vee x_3,$$
$$c_2 = \neg x_2 \vee \neg x_3 \vee x_4$$

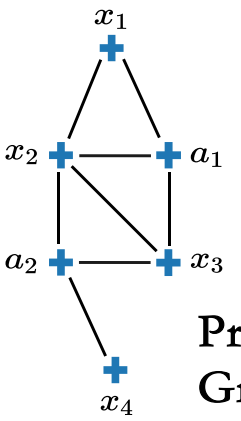
3-SAT Problem

2

$$H_C(X, A) = 3 + x_1 + 3x_2 - 2x_3 - x_4 - 2a_2 + x_1x_2 - x_2x_3 - 2a_1x_1 - 2a_1x_2 + a_1x_3 - 2a_2x_2 + 2a_2x_3 + a_2x_4$$

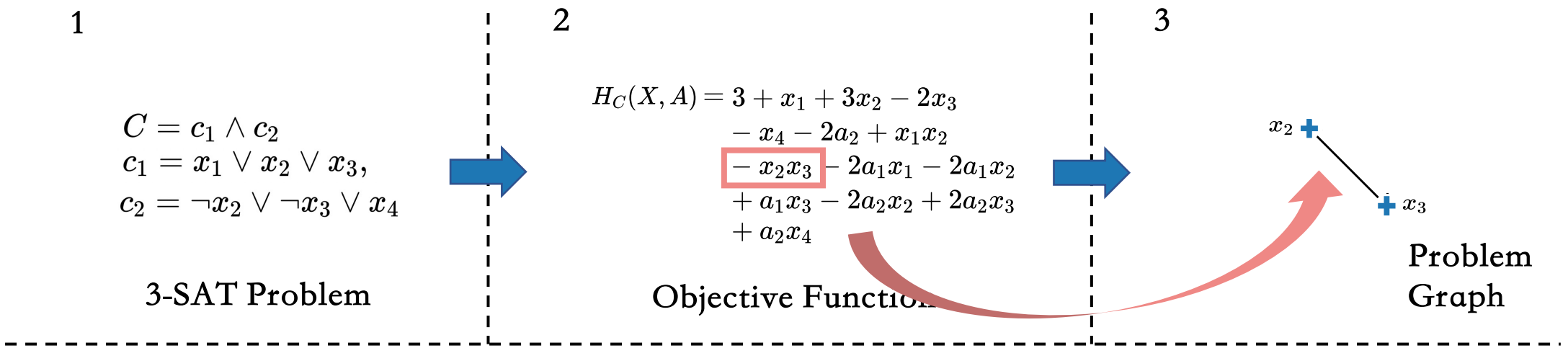
Objective Function

3



Problem Graph

Deploying 3-SAT Problem to Quantum Annealer



Deploying 3-SAT Problem to Quantum Annealer

1

$$C = c_1 \wedge c_2$$
$$c_1 = x_1 \vee x_2 \vee x_3,$$
$$c_2 = \neg x_2 \vee \neg x_3 \vee x_4$$

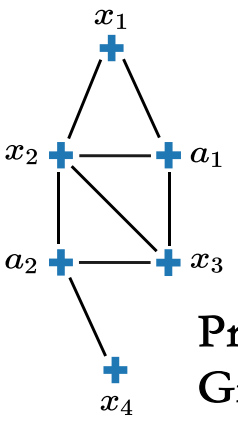
3-SAT Problem

2

$$H_C(X, A) = 3 + x_1 + 3x_2 - 2x_3 - x_4 - 2a_2 + x_1x_2 - x_2x_3 - 2a_1x_1 - 2a_1x_2 + a_1x_3 - 2a_2x_2 + 2a_2x_3 + a_2x_4$$

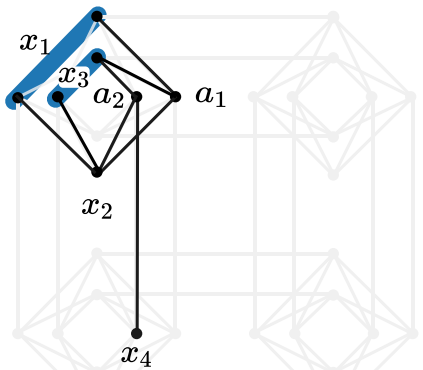
Objective Function

3



Problem Graph

4



Processor Topology

Embedding

Deploying 3-SAT Problem to Quantum Annealer

1

$$C = c_1 \wedge c_2$$
$$c_1 = x_1 \vee x_2 \vee x_3,$$
$$c_2 = \neg x_2 \vee \neg x_3 \vee x_4$$

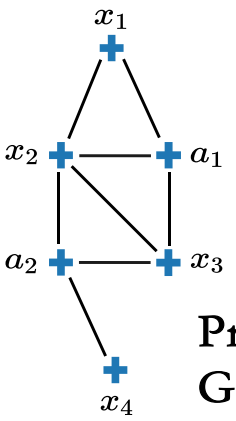
3-SAT Problem

2

$$H_C(X, A) = 3 + x_1 + 3x_2 - 2x_3 - x_4 - 2a_2 + x_1x_2 - x_2x_3 - 2a_1x_1 - 2a_1x_2 + a_1x_3 - 2a_2x_2 + 2a_2x_3 + a_2x_4$$

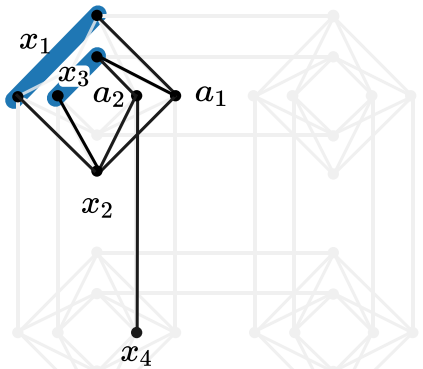
Objective Function

3



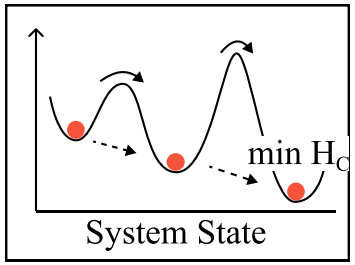
Problem Graph

4



Embedding

5



Quantum Annealing

Deploying 3-SAT Problem to Quantum Annealer

1

$$C = c_1 \wedge c_2$$
$$c_1 = x_1 \vee x_2 \vee x_3,$$
$$c_2 = \neg x_2 \vee \neg x_3 \vee x_4$$

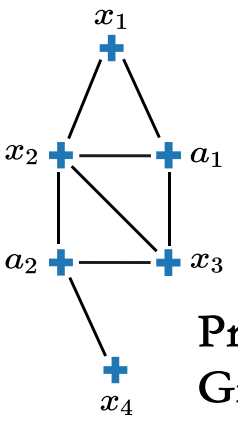
3-SAT Problem

2

$$H_C(X, A) = 3 + x_1 + 3x_2 - 2x_3 - x_4 - 2a_2 + x_1x_2 - x_2x_3 - 2a_1x_1 - 2a_1x_2 + a_1x_3 - 2a_2x_2 + 2a_2x_3 + a_2x_4$$

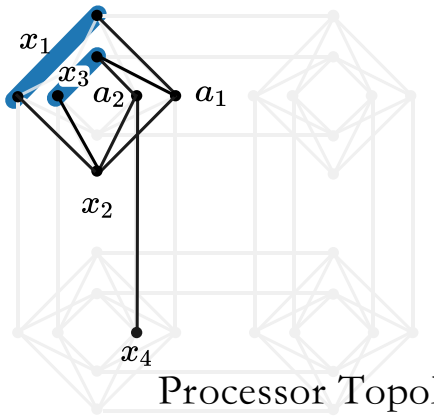
Objective Function

3



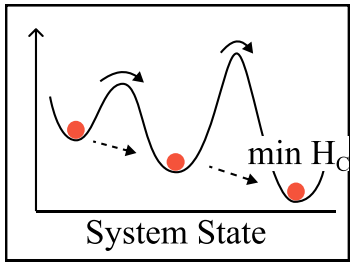
Problem Graph

4



Embedding

5



Quantum Annealing

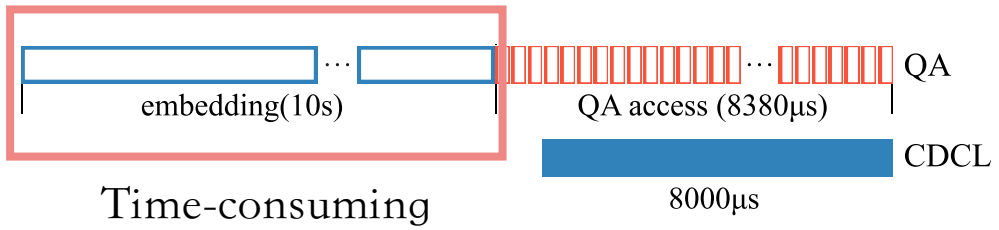
6

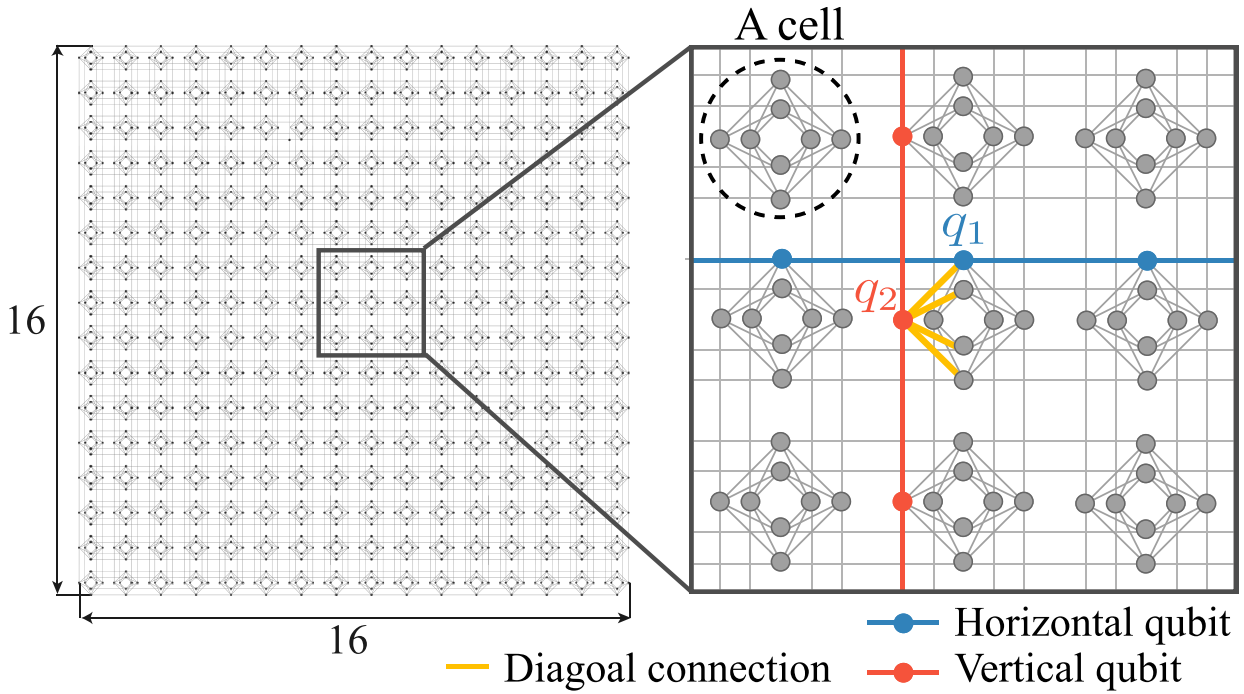
$$min\ H_c(X, A) = 0$$

$$x_1 = x_2 = 0$$
$$x_3 = x_4 = 1$$

Solution

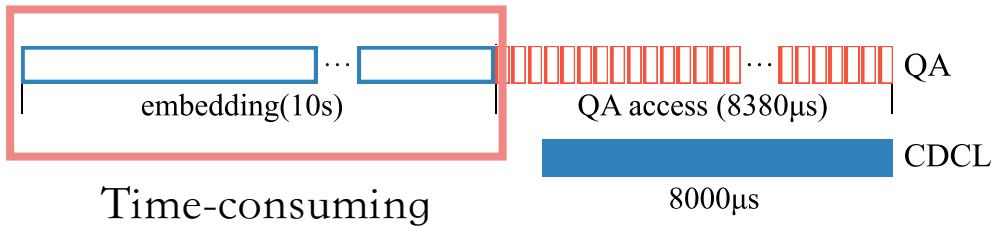
Challenge 1 of QA: High embedding latency

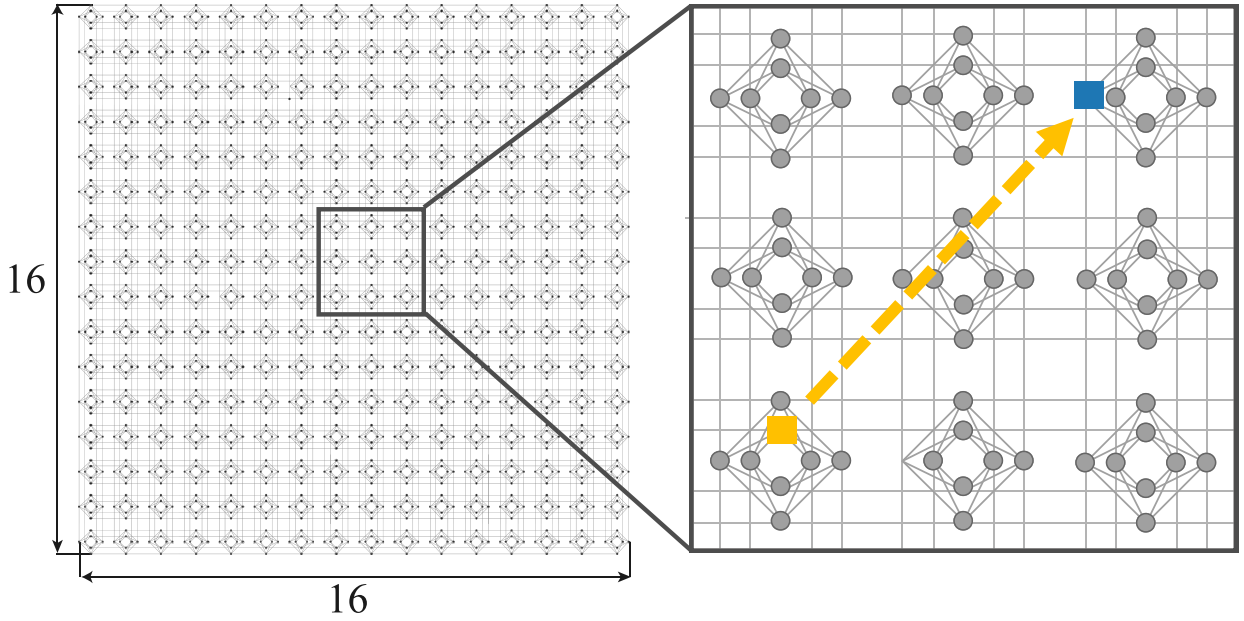




Processor topology of the D-Wave 2000Q

Challenge 1 of QA: High embedding latency

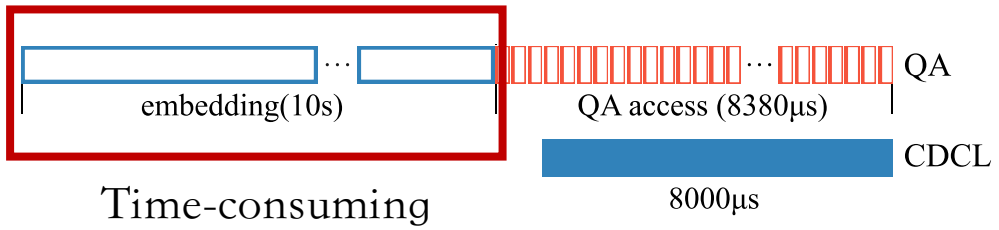


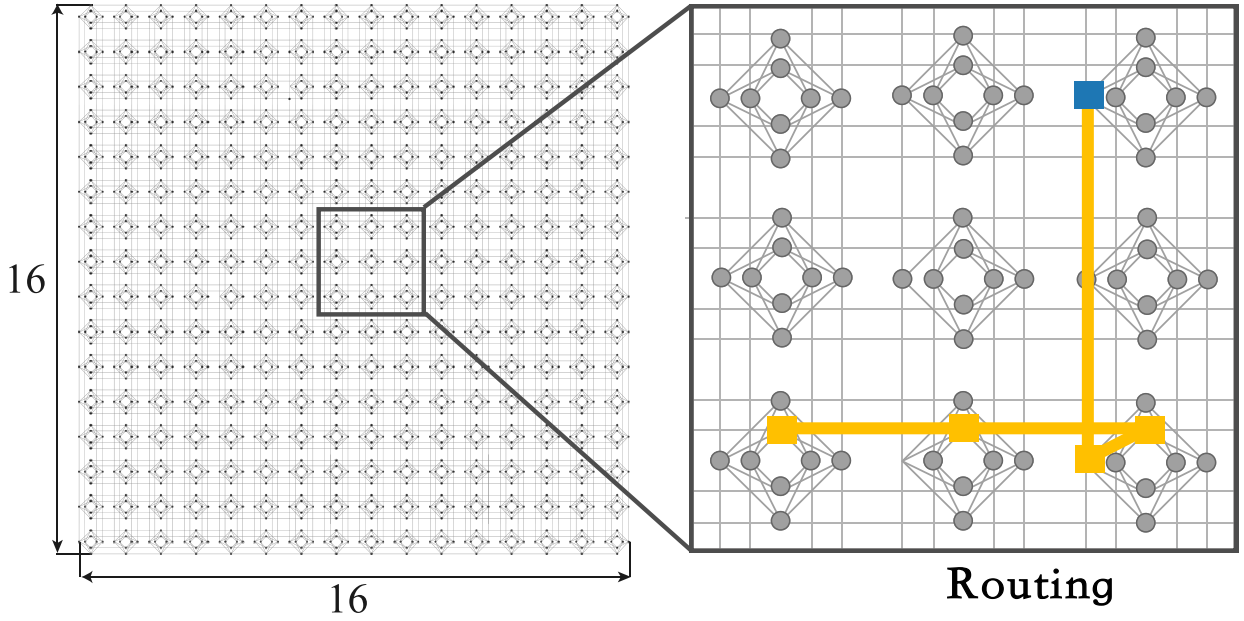


The two most time-consuming parts of the previous embedding schemes: **routing, adjustment**.

ZHEJIANG UNIVERSITY

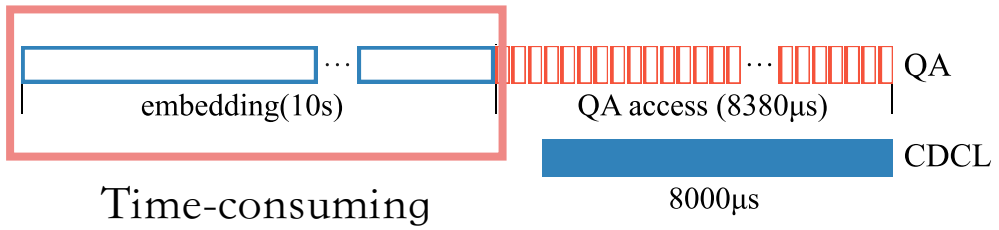
Challenge 1 of QA: High compilation latency

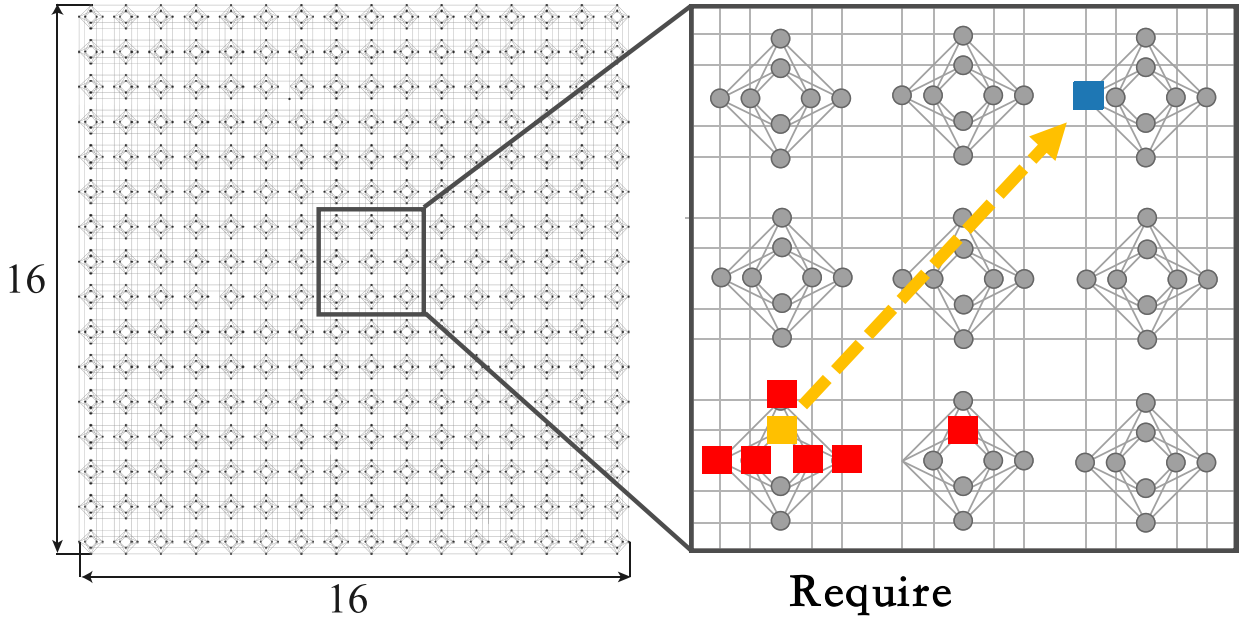




The two most time-consuming parts of the previous embedding schemes: **routing, adjustment**.

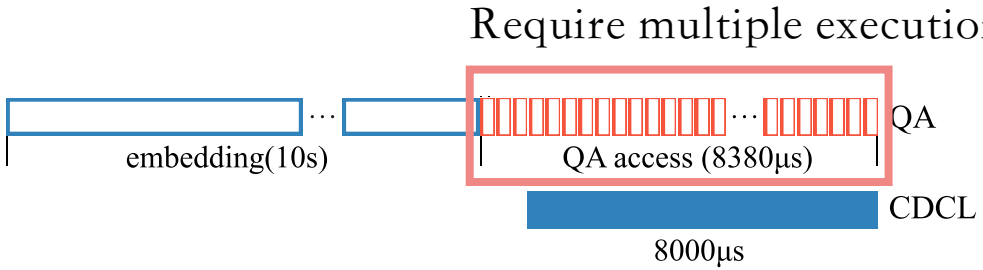
Challenge 1 of QA: High compilation latency





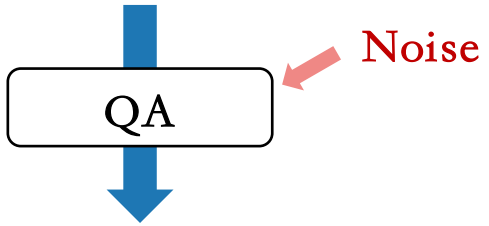
The two most time-consuming parts of the previous embedding schemes: **routing, adjustment**.

Challenge 2 of QA: Noise



50 executions to find a solution for a 50-variable problem.

$$C = c_1 \wedge c_2$$
$$c_1 = x_1 \vee x_2 \vee x_3,$$
$$c_2 = \neg x_2 \vee \neg x_3 \vee x_4$$



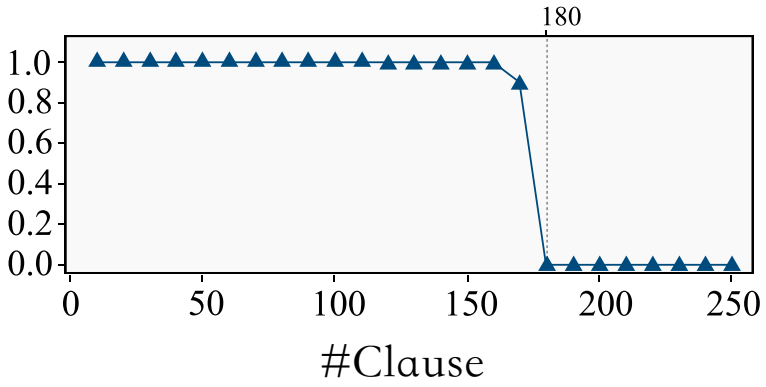
$$x_1 = x_2 = 0$$
$$x_3 = x_4 = 1$$

Some variables get wrong assignment.

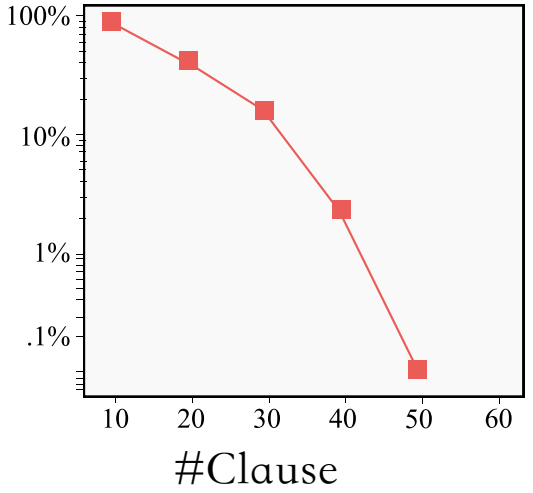
Challenge 3 of QA: Limited Problem Scale

Limited by both the embedding and noise.

Successful rate of embedding

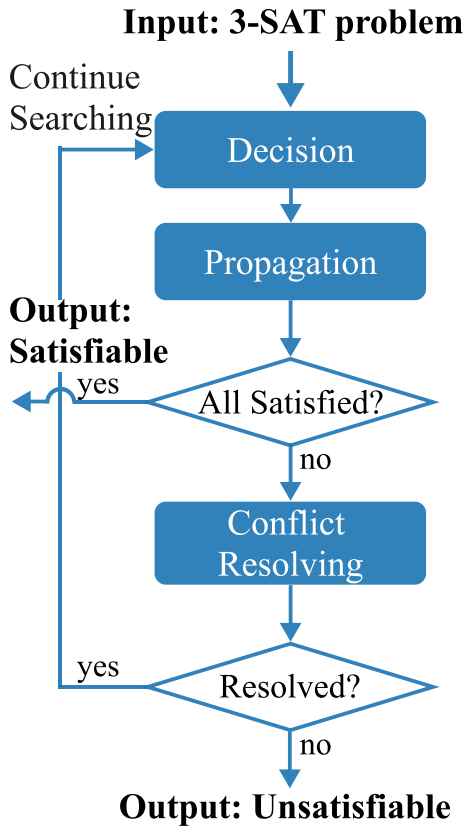


Success rate of finding solution



The success rate decreases exponentially as the problem size increases.

HyQSAT Workflow

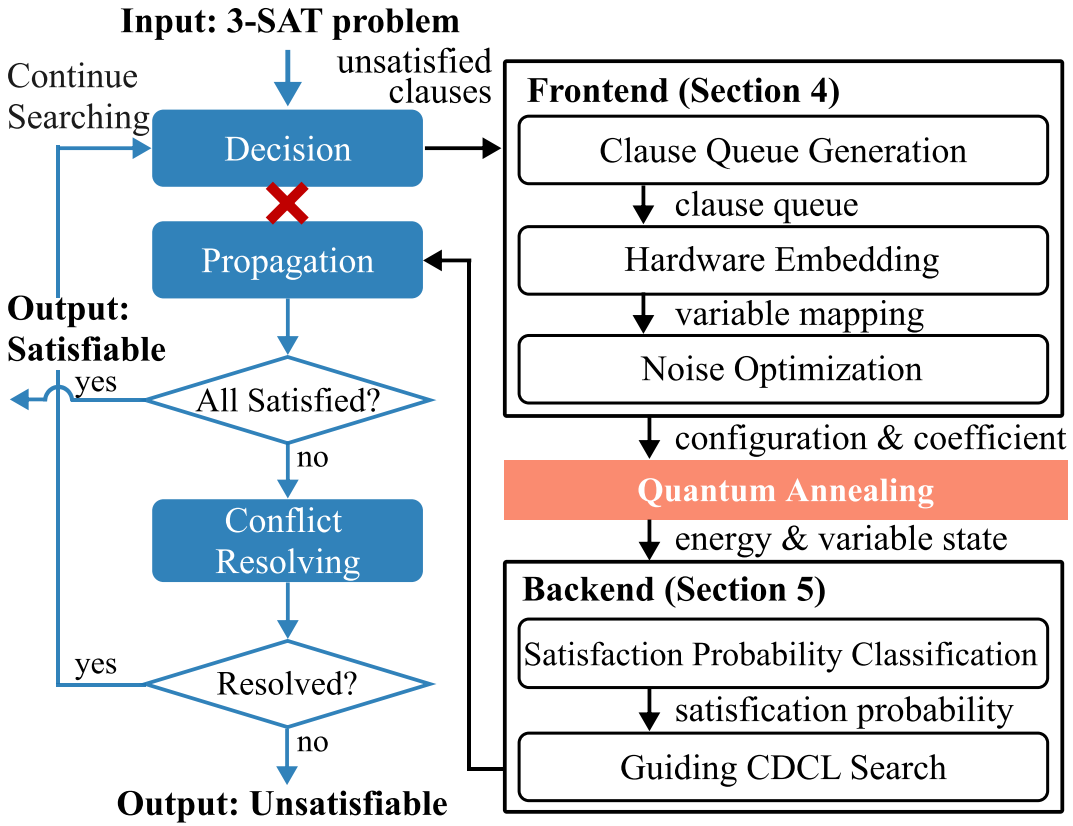


CDCL workflow

CDCL	Quantum Annealing
Large scale	Small scale
Difficulty in solving 'hard' clauses	Quantum speedup;
8000μs	10s+120μs

Embedding

HyQSAT Workflow



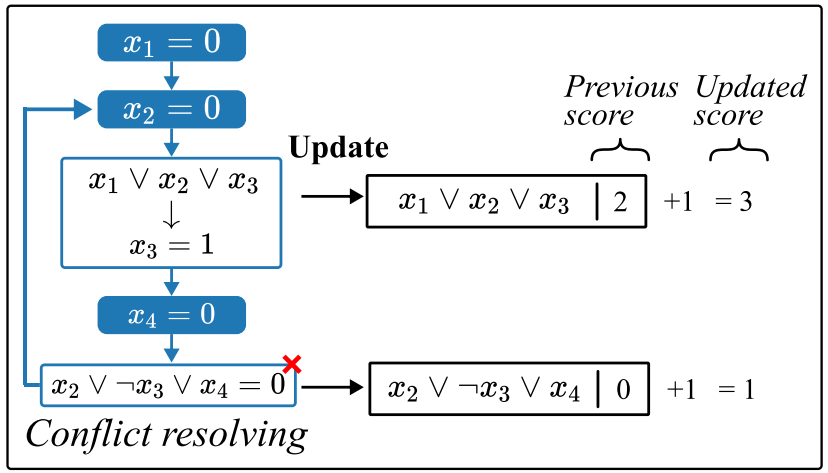
HyQSAT workflow

CDCL	Quantum Annealing
Large scale	Small scale
Difficulty in solving 'hard' clauses	Quantum speedup;
8000μs	10s+120μs

Move critical clauses from CDCL to annealing:

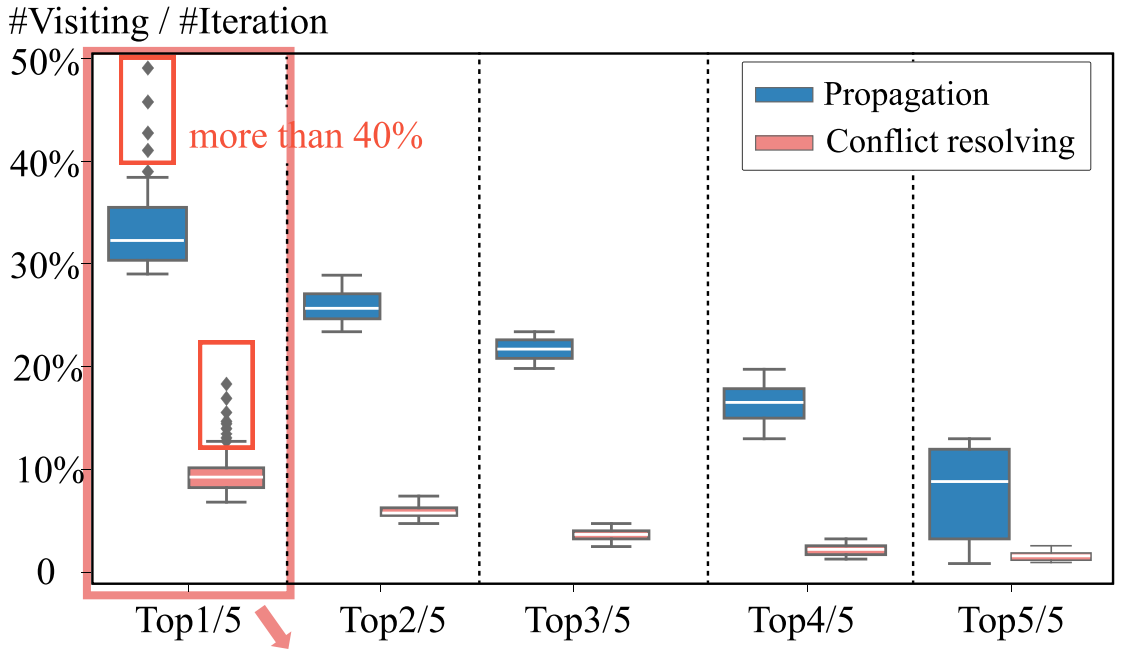
1. difficult for CDCL
2. efficiently embedded for quantum annealer

HyQSAT Frontend: Identify Difficult Clauses



(a) Activity score updating

Measure the difficulty of clauses, predicting their visiting frequency.

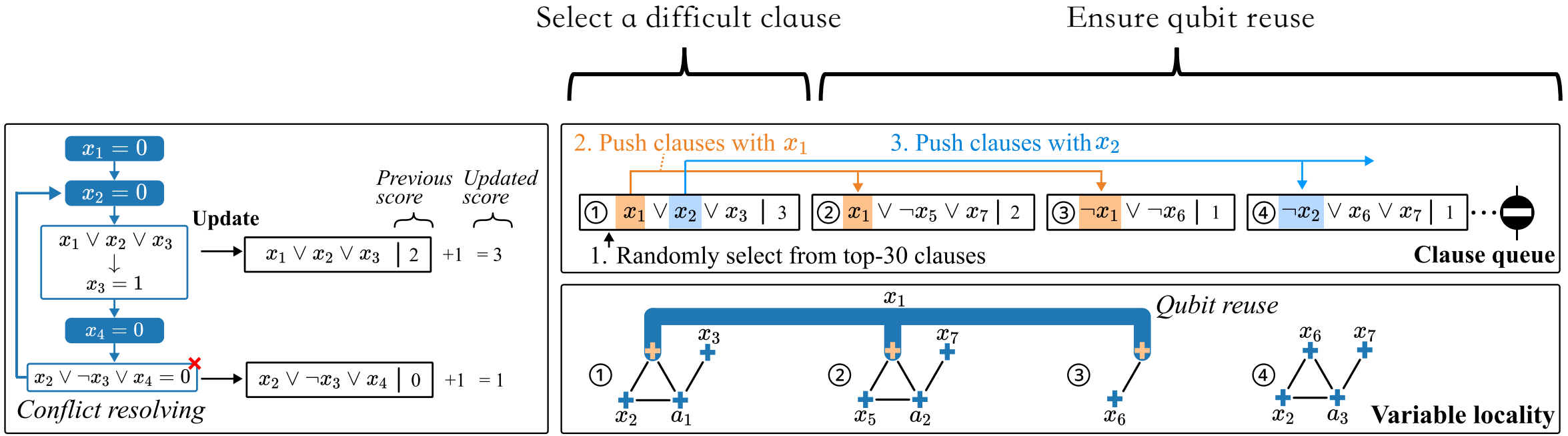


Difficult clauses

Clause visiting frequency = Difficulty

ZHEJIANG UNIVERSITY

HyQSAT Frontend: Identify Difficult Clauses

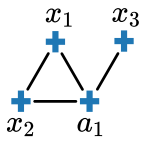
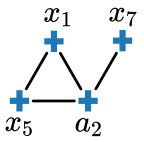
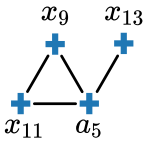


Measure the difficulty of clauses, predicting their visiting frequency.

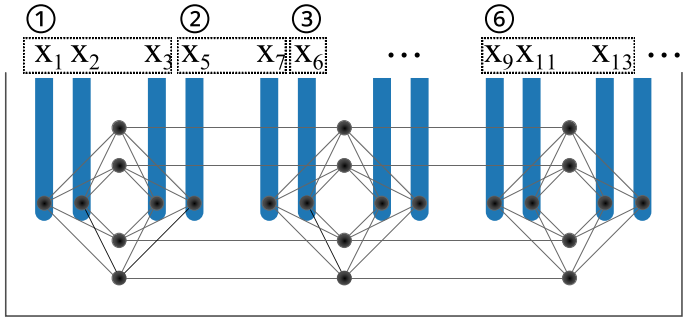
ZHEJIANG UNIVERSITY

HyQSAT Frontend: Fast Embedding

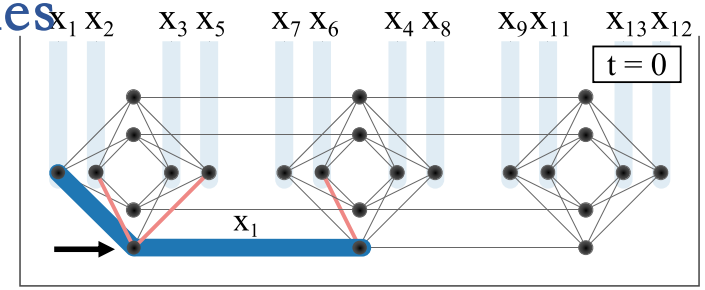
Step 1: Allocate variables to qubits of vertical lines according to their order in the clause queue

Clause:	① $x_1 \vee x_2 \vee x_3$	② $x_1 \vee \neg x_5 \vee x_7$	③ $\neg x_1 \vee \neg x_6$	⑥ $x_9 \vee x_{11} \vee x_{13}$
Problem graph:				
Connection requirement list:	x_1:{x_2} a_1:{x_1,x_2, x_3}	x_1:{x_2,x_5} a_1:{x_1,x_2,x_3} a_2:{x_1,x_5,x_7}	x_1:{x_2,x_5,x_6} a_1:{x_1,x_2,x_3} a_2:{x_1,x_5,x_7}	x_1:{x_2,x_5,x_6} a_1:{x_1,x_2,x_3} a_2:{x_1,x_5,x_7} … a_5:{x_9,x_{11},x_{13}}

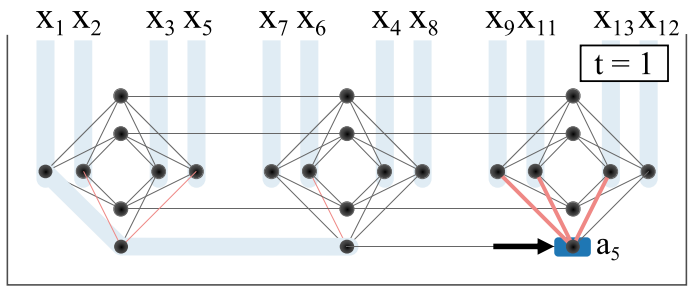
gathered for allocation together



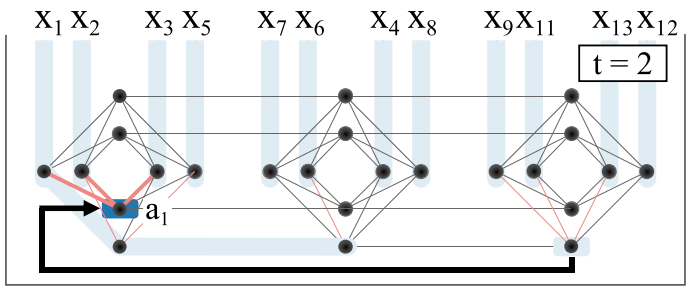
Step 2: Allocate variables to qubits of horizontal lines



x_1 :{x_2, x_5, x_6}



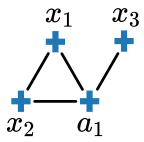
a_5 :{x_9, x_{11}, x_{13}}



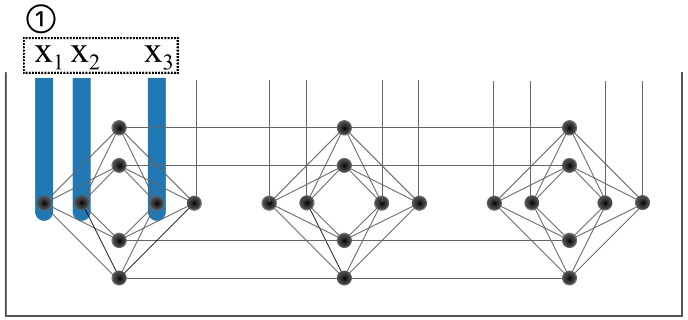
a_1 :{x_1, x_2, x_3}

HyQSAT Frontend: Fast Embedding

Step 1: Allocate variables to qubits of vertical lines

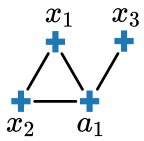
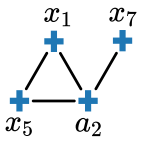
Clause:	① $x_1 \vee x_2 \vee x_3$
Problem graph:	
Connection requirement list:	x_1:{x_2} a_1:{x_1,x_2, x_3}

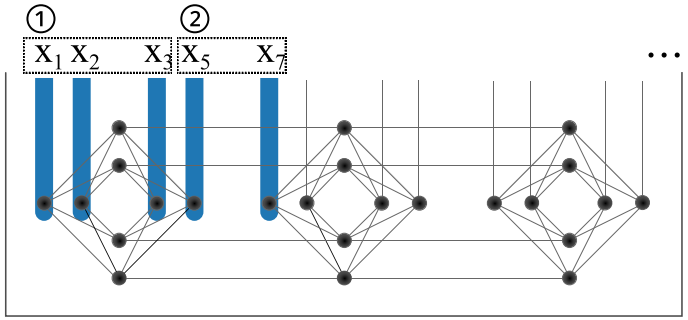
gathered for allocation together



HyQSAT Frontend: Fast Embedding

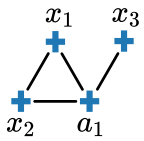
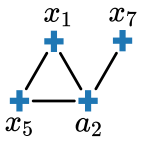
Step 1: Allocate variables to qubits of **vertical lines**

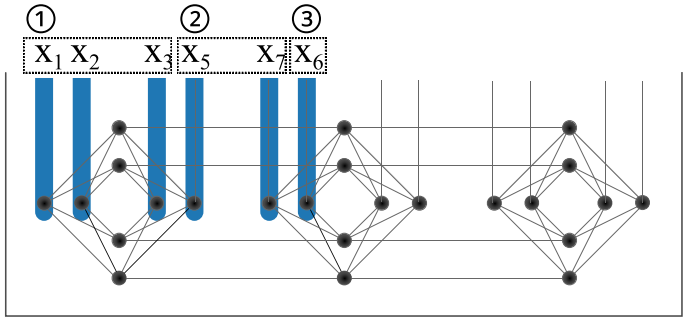
Clause:	① $x_1 \vee x_2 \vee x_3$	② $x_1 \vee \neg x_5 \vee x_7$
Problem graph:		
Connection requirement list:	x_1:{x_2} a_1:{x_1,x_2, x_3}	x_1:{x_2,x_5} a_1:{x_1,x_2,x_3} a_2:{x_1,x_5,x_7}



HyQSAT Frontend: Fast Embedding

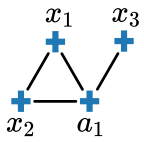
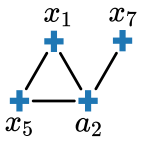
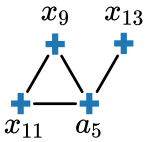
Step 1: Allocate variables to qubits of vertical lines

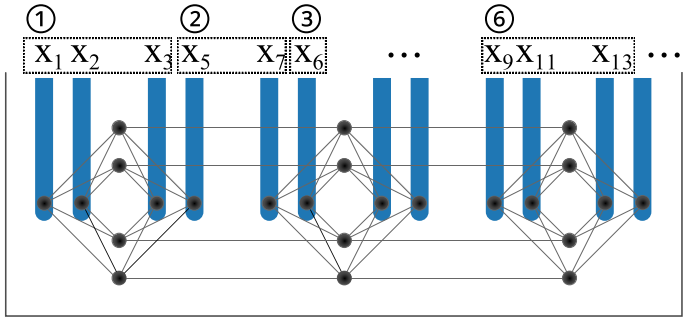
Clause:	① $x_1 \vee x_2 \vee x_3$	② $x_1 \vee \neg x_5 \vee x_7$	③ $\neg x_1 \vee \neg x_6$
Problem graph:			
Connection requirement list:	x_1:{x_2} a_1:{x_1,x_2, x_3}	x_1:{x_2,x_5} a_1:{x_1,x_2,x_3} a_2:{x_1,x_5,x_7}	x_1:{x_2,x_5,x_6} a_1:{x_1,x_2,x_3} a_2:{x_1,x_5,x_7}



HyQSAT Frontend: Fast Embedding

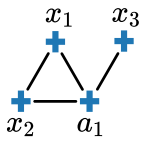
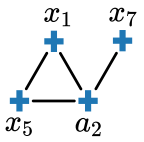
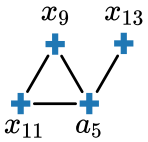
Step 1: Allocate variables to qubits of vertical lines

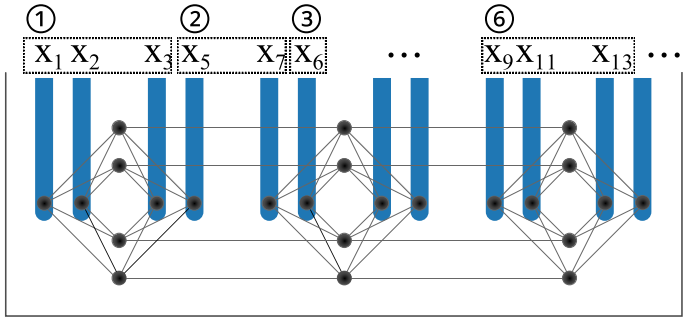
Clause:	① $x_1 \vee x_2 \vee x_3$	② $x_1 \vee \neg x_5 \vee x_7$	③ $\neg x_1 \vee \neg x_6$	⑥ $x_9 \vee x_{11} \vee x_{13}$
Problem graph:				
Connection requirement list:	x_1:{x_2} a_1:{x_1,x_2, x_3}	x_1:{x_2,x_5} a_1:{x_1,x_2,x_3} a_2:{x_1,x_5,x_7}	x_1:{x_2,x_5,x_6} a_1:{x_1,x_2,x_3} a_2:{x_1,x_5,x_7}	x_1:{x_2,x_5,x_6} a_1:{x_1,x_2,x_3} a_2:{x_1,x_5,x_7} … a_5:{x_9,x_{11},x_{13}}



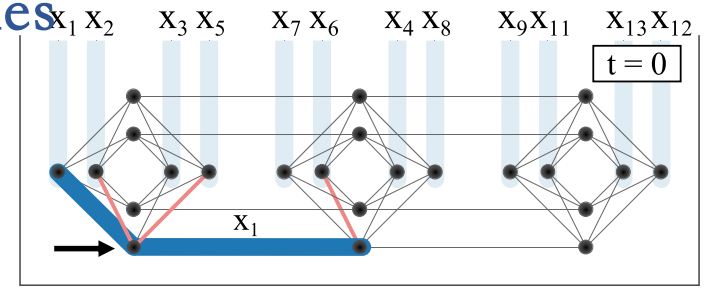
HyQSAT Frontend: Fast Embedding

Step 1: Allocate variables to qubits of vertical lines

Clause:	① $x_1 \vee x_2 \vee x_3$	② $x_1 \vee \neg x_5 \vee x_7$	③ $\neg x_1 \vee \neg x_6$	⑥ $x_9 \vee x_{11} \vee x_{13}$
Problem graph:				
Connection requirement list:	x_1:{x_2} a_1:{x_1,x_2, x_3}	x_1:{x_2,x_5} a_1:{x_1,x_2,x_3} a_2:{x_1,x_5,x_7}	x_1:{x_2,x_5,x_6} a_1:{x_1,x_2,x_3} a_2:{x_1,x_5,x_7}	x_1:{x_2,x_5,x_6} a_1:{x_1,x_2,x_3} a_2:{x_1,x_5,x_7} … a_5:{x_9,x_{11},x_{13}}



Step 2: Allocate variables to qubits of horizontal lines

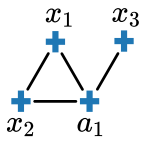
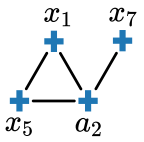
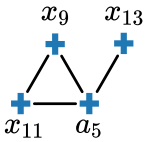


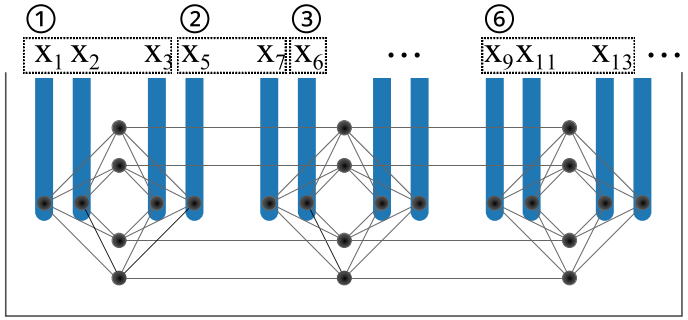
x_1 :{x_2, x_5, x_6}

ZHEJIANG UNIVERSITY

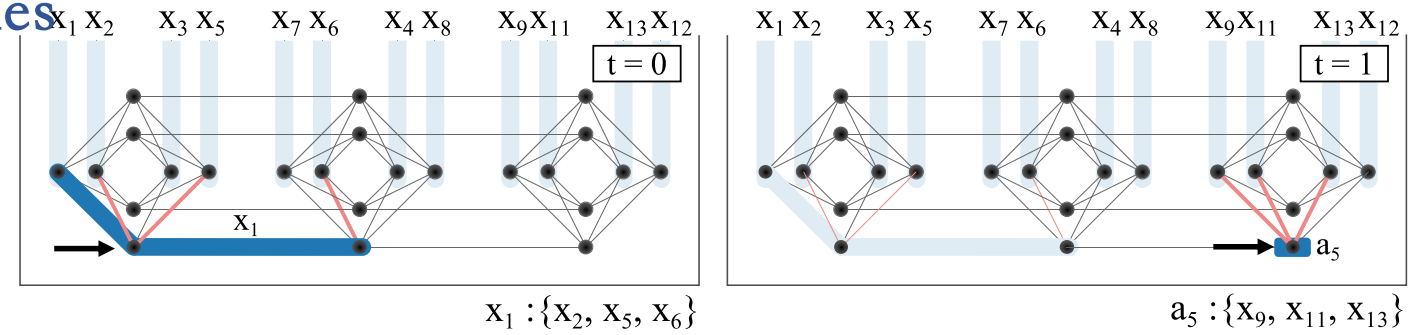
HyQSAT Frontend: Fast Embedding

Step 1: Allocate variables to qubits of vertical lines

Clause:	① $x_1 \vee x_2 \vee x_3$	② $x_1 \vee \neg x_5 \vee x_7$	③ $\neg x_1 \vee \neg x_6$	⑥ $x_9 \vee x_{11} \vee x_{13}$
Problem graph:				
Connection requirement list:	x_1:{x_2} a_1:{x_1,x_2, x_3}	x_1:{x_2,x_5} a_1:{x_1,x_2,x_3} a_2:{x_1,x_5,x_7}	x_1:{x_2,x_5,x_6} a_1:{x_1,x_2,x_3} a_2:{x_1,x_5,x_7}	x_1:{x_2,x_5,x_6} a_1:{x_1,x_2,x_3} a_2:{x_1,x_5,x_7} … a_5:{x_9,x_{11},x_{13}}



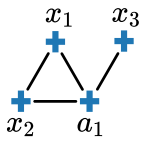
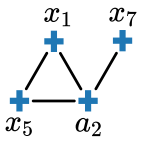
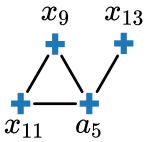
Step 2: Allocate variables to qubits of horizontal lines

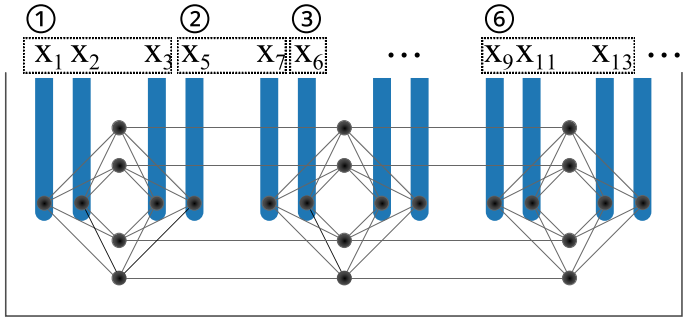


ZHEJIANG UNIVERSITY

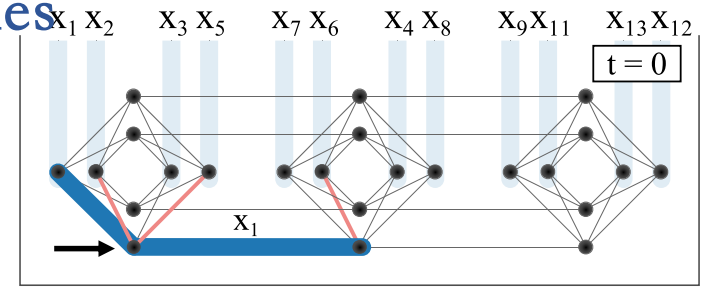
HyQSAT Frontend: Fast Embedding

Step 1: Allocate variables to qubits of vertical lines

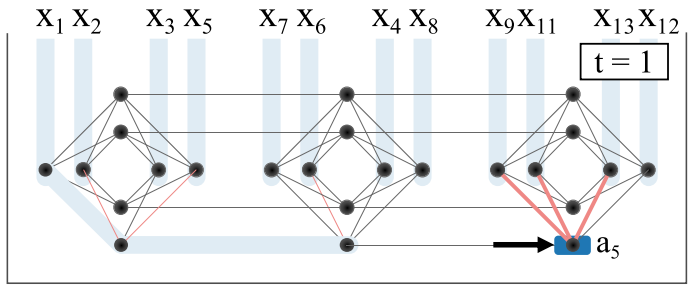
Clause:	① $x_1 \vee x_2 \vee x_3$	② $x_1 \vee \neg x_5 \vee x_7$	③ $\neg x_1 \vee \neg x_6$	⑥ $x_9 \vee x_{11} \vee x_{13}$
Problem graph:				
Connection requirement list:	x_1:{x_2} a_1:{x_1,x_2, x_3}	x_1:{x_2,x_5} a_1:{x_1,x_2,x_3} a_2:{x_1,x_5,x_7}	x_1:{x_2,x_5,x_6} a_1:{x_1,x_2,x_3} a_2:{x_1,x_5,x_7}	x_1:{x_2,x_5,x_6} a_1:{x_1,x_2,x_3} a_2:{x_1,x_5,x_7} … a_5:{x_9,x_{11},x_{13}}



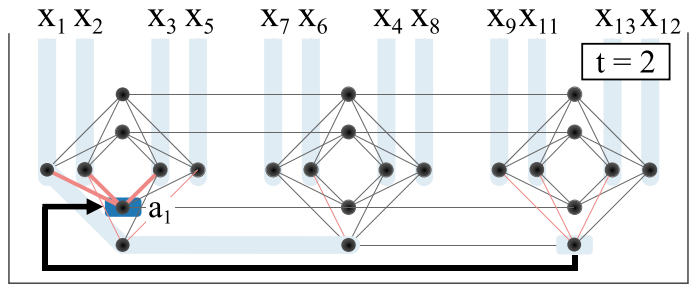
Step 2: Allocate variables to qubits of horizontal lines



x_1 :{x_2, x_5, x_6}

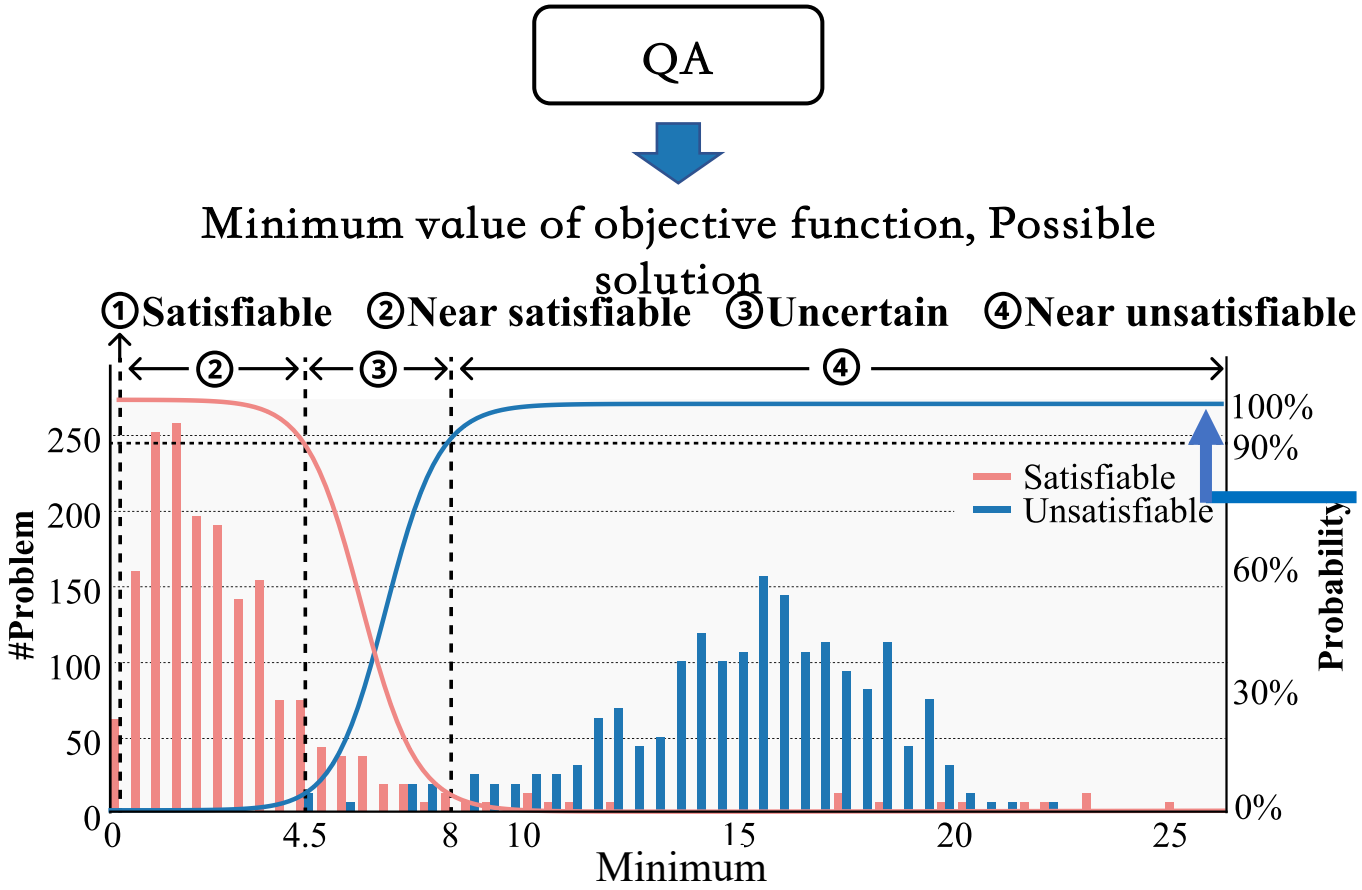


a_5 :{x_9, x_{11}, x_{13}}



a_1 :{x_1, x_2, x_3}

HyQSAT Backend: Satisfaction Probability Classification



Based on the noise model of D-Wave 2000Q

Gaussian Naive Bayes model to estimate the probability of satisfaction

① **Satisfiable problem:** [0, 0].
② **Near satisfiable problem:** (0, 4.5].
③ **Uncertain problem:** (4.5, 8].
④ **Near unsatisfiable problem:** (8, +∞].

HyQSAT Backend: Guiding CDCL Search

Depending on **the number of embedded clauses and their satisfaction probability**, we divide them into **four cases** and propose several feedback strategies to prune the CDCL search space.

	Satisfiable	Near satisfiable	Uncertain	Near unsatisfiable
All embedded	Strategy 1	Strategy 2	Strategy 3	Strategy 4
Not all embedded				

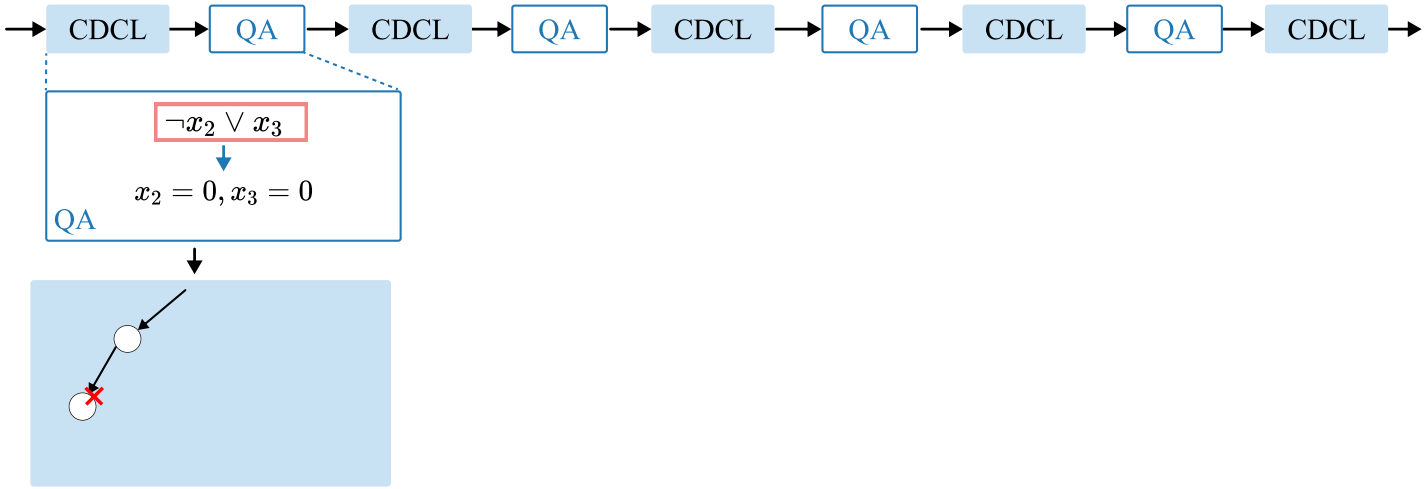
Contribute to no acceleration for the classic CDCL.

HyQSAT Backend: Guiding CDCL Search

Depending on **the number of embedded clauses and their satisfaction probability**, we divide them into **four cases** and propose several feedback strategies to prune the CDCL search space.

	Satisfiable	Near satisfiable	Uncertain	Near unsatisfiable
All embedded	Strategy 1	Strategy 2	Strategy 3	Strategy 4
Not all embedded				

Feedback strategy 1, 2, 4



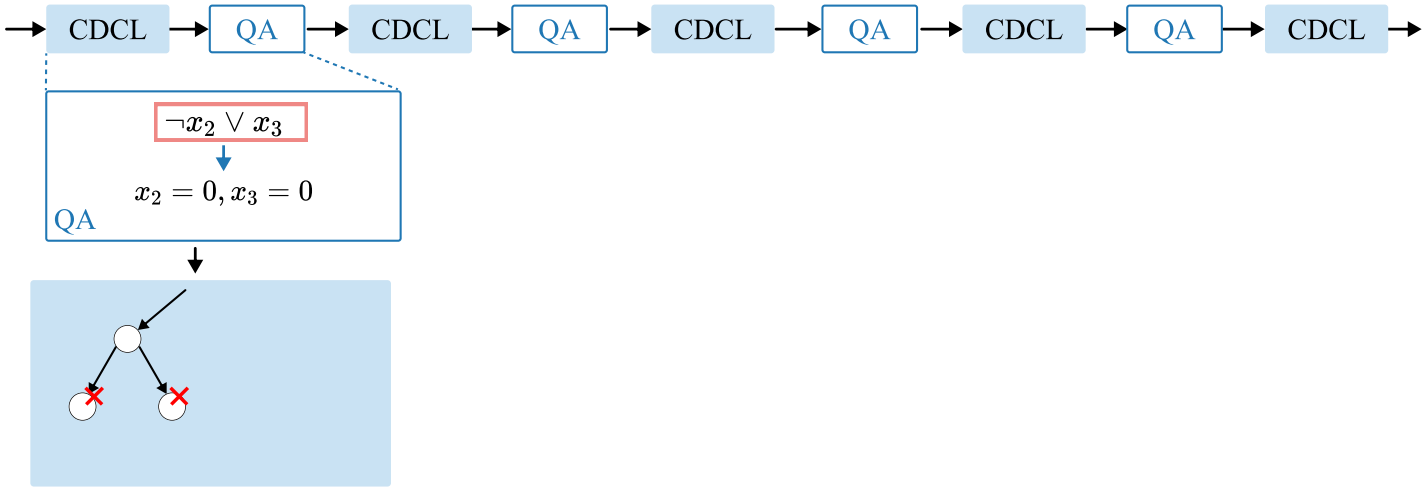
Strategy 1

HyQSAT Backend: Guiding CDCL Search

Depending on **the number of embedded clauses and their satisfaction probability**, we divide them into **four cases** and propose several feedback strategies to prune the CDCL search space.

	Satisfiable	Near satisfiable	Uncertain	Near unsatisfiable
All embedded	Strategy 1	Strategy 2	Strategy 3	Strategy 4
Not all embedded				

Feedback strategy 1, 2, 4



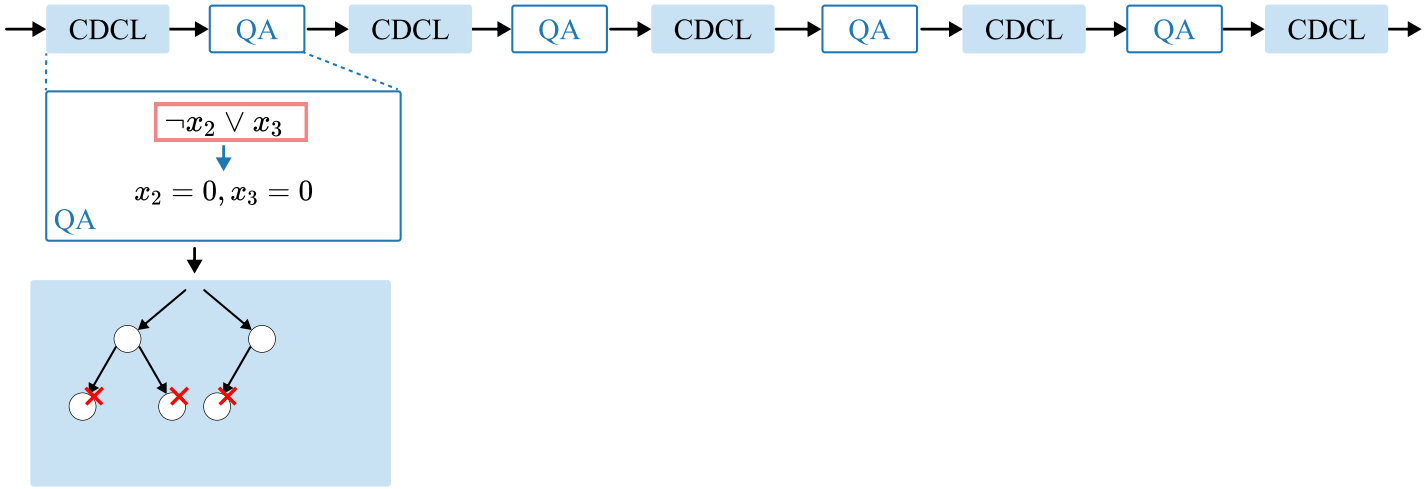
Strategy 1

HyQSAT Backend: Guiding CDCL Search

Depending on **the number of embedded clauses and their satisfaction probability**, we divide them into **four cases** and propose several feedback strategies to prune the CDCL search space.

	Satisfiable	Near satisfiable	Uncertain	Near unsatisfiable
All embedded	Strategy 1	Strategy 2	Strategy 3	Strategy 4
Not all embedded				

Feedback strategy 1, 2, 4



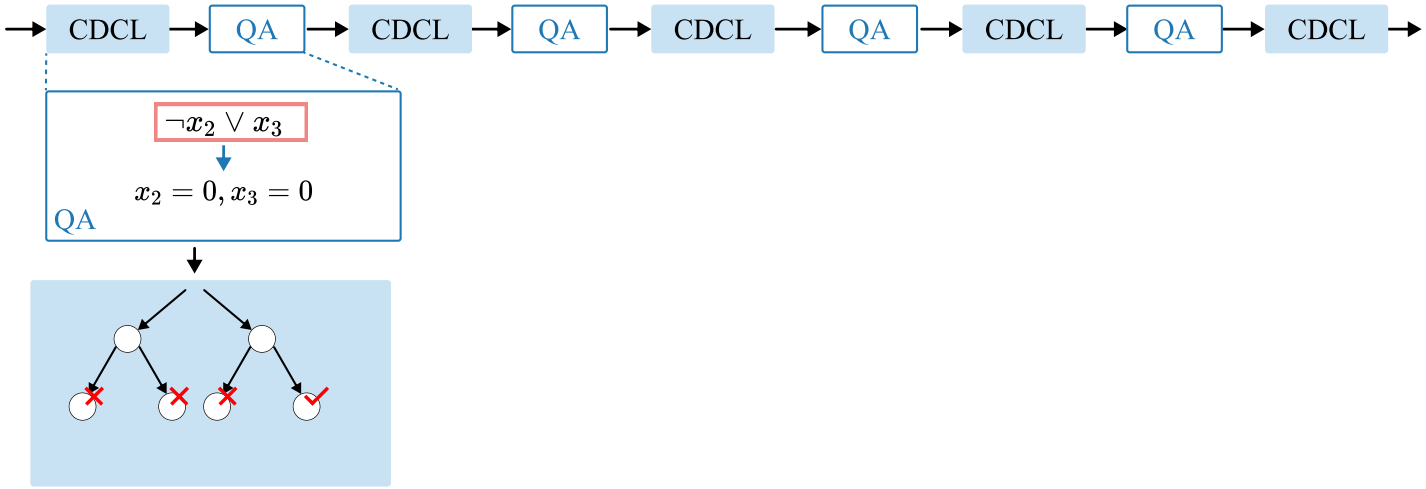
Strategy 1

HyQSAT Backend: Guiding CDCL Search

Depending on **the number of embedded clauses and their satisfaction probability**, we divide them into **four cases** and propose several feedback strategies to prune the CDCL search space.

	Satisfiable	Near satisfiable	Uncertain	Near unsatisfiable
All embedded	Strategy 1	Strategy 2	Strategy 3	Strategy 4
Not all embedded				

Feedback strategy 1, 2, 4



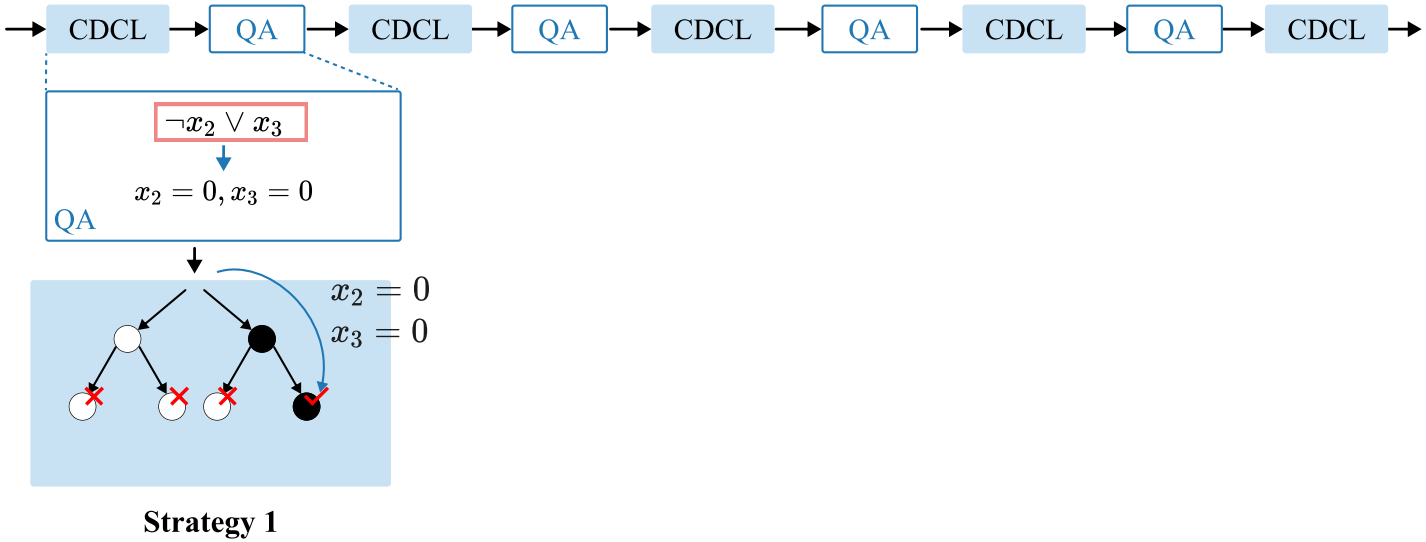
Strategy 1

HyQSAT Backend: Guiding CDCL Search

Depending on **the number of embedded clauses and their satisfaction probability**, we divide them into **four cases** and propose several feedback strategies to prune the CDCL search space.

	Satisfiable	Near satisfiable	Uncertain	Near unsatisfiable
All embedded	Strategy 1	Strategy 2	Strategy 3	Strategy 4
Not all embedded				

Feedback strategy 1, 2, 4



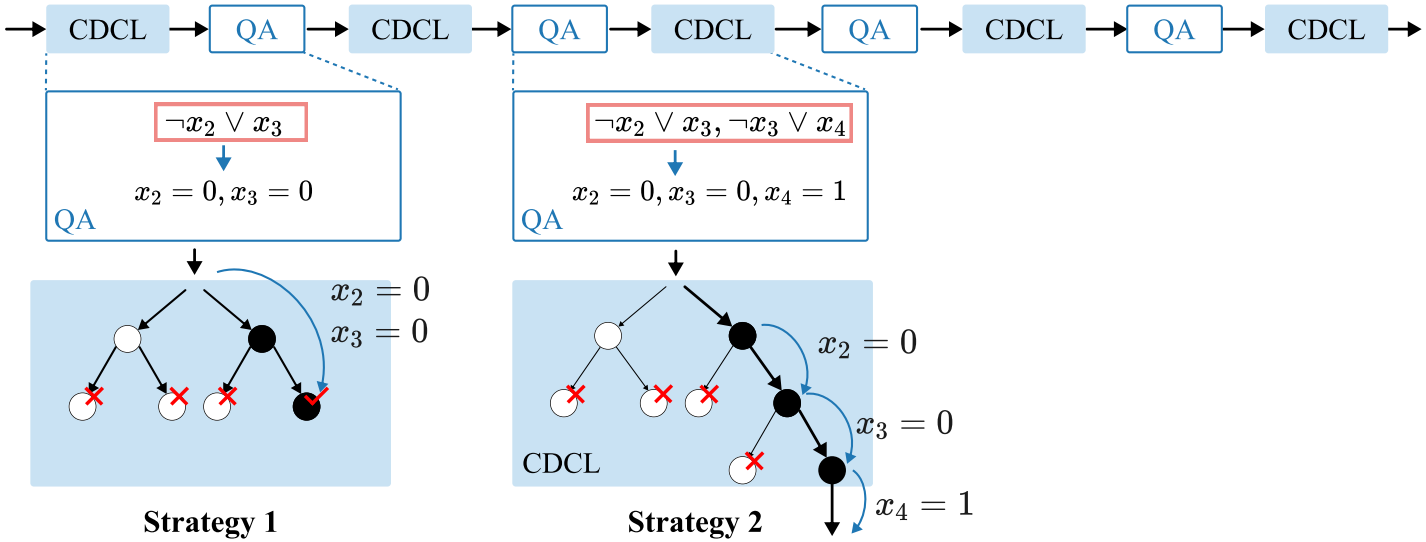
Strategy 1

HyQSAT Backend: Guiding CDCL Search

Depending on **the number of embedded clauses and their satisfaction probability**, we divide them into **four cases** and propose several feedback strategies to prune the CDCL search space.

	Satisfiable	Near satisfiable	Uncertain	Near unsatisfiable
All embedded	Strategy 1	Strategy 2	Strategy 3	Strategy 4
Not all embedded				

Feedback strategy 1, 2, 4

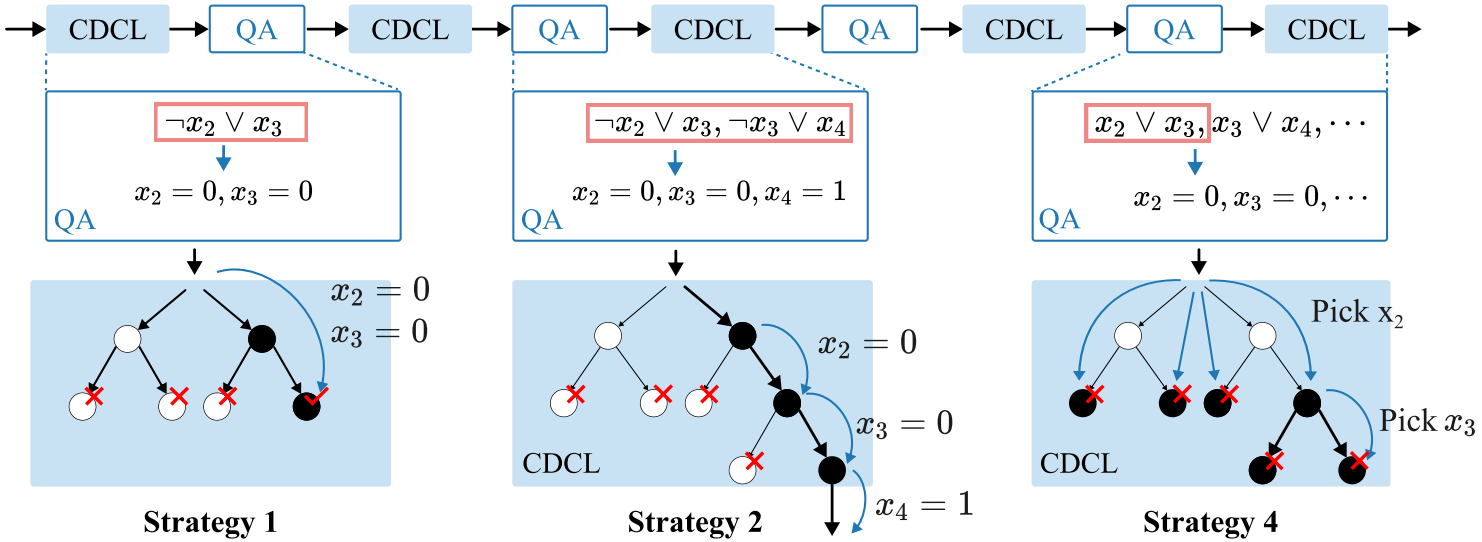


HyQSAT Backend: Guiding CDCL Search

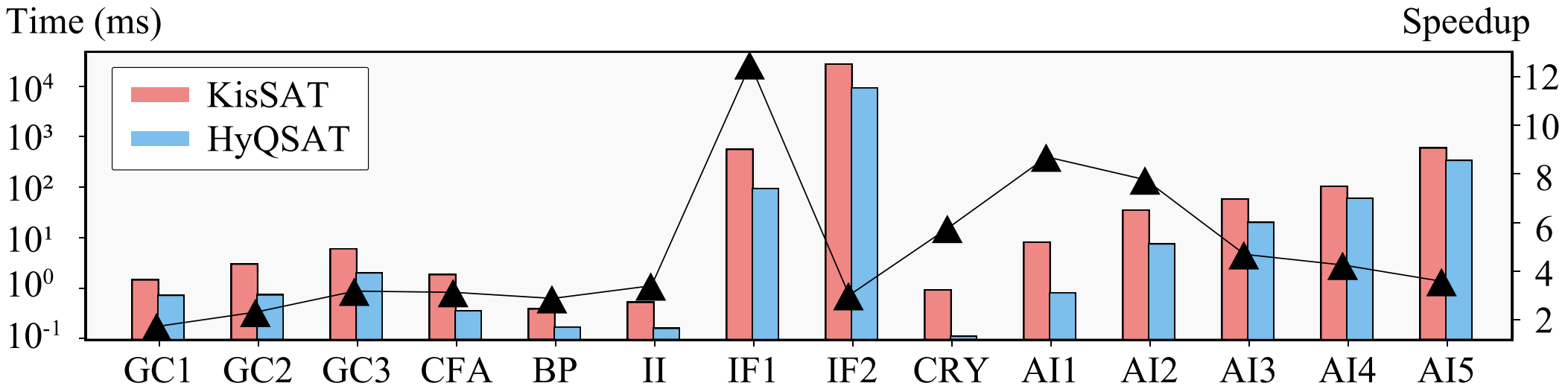
Depending on **the number of embedded clauses and their satisfaction probability**, we divide them into **four cases** and propose several feedback strategies to prune the CDCL search space.

	Satisfiable	Near satisfiable	Uncertain	Near unsatisfiable
All embedded	Strategy 1	Strategy 2	Strategy 3	Strategy 4
Not all embedded				

Feedback strategy 1, 2, 4



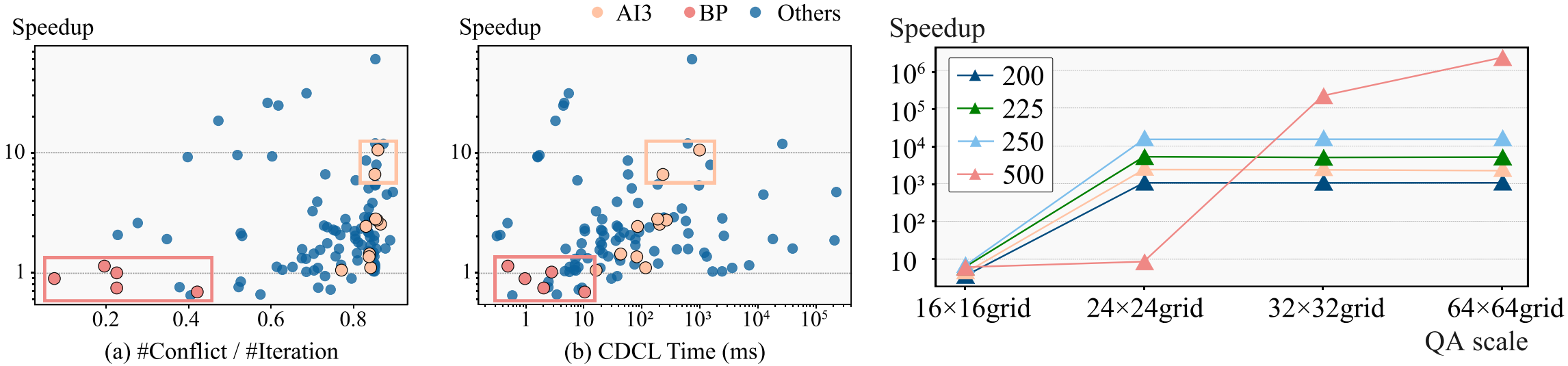
Evaluation: Speedup on the Real-World Quantum Annealer



graph coloring (CG), circuit fault analysis (CFA), block planning (BP), inductive inference (II), integer factorization (IF), cryptography (CRY), and artificial intelligence (AI)

- 7 domains, 11 benchmarks
- D-Wave 2000Q real-world quantum annealer
- 4.92X speedup compared to KisSAT (win SAT competition 2022)

Evaluation: Speedup on the Real-World Quantum Annealer

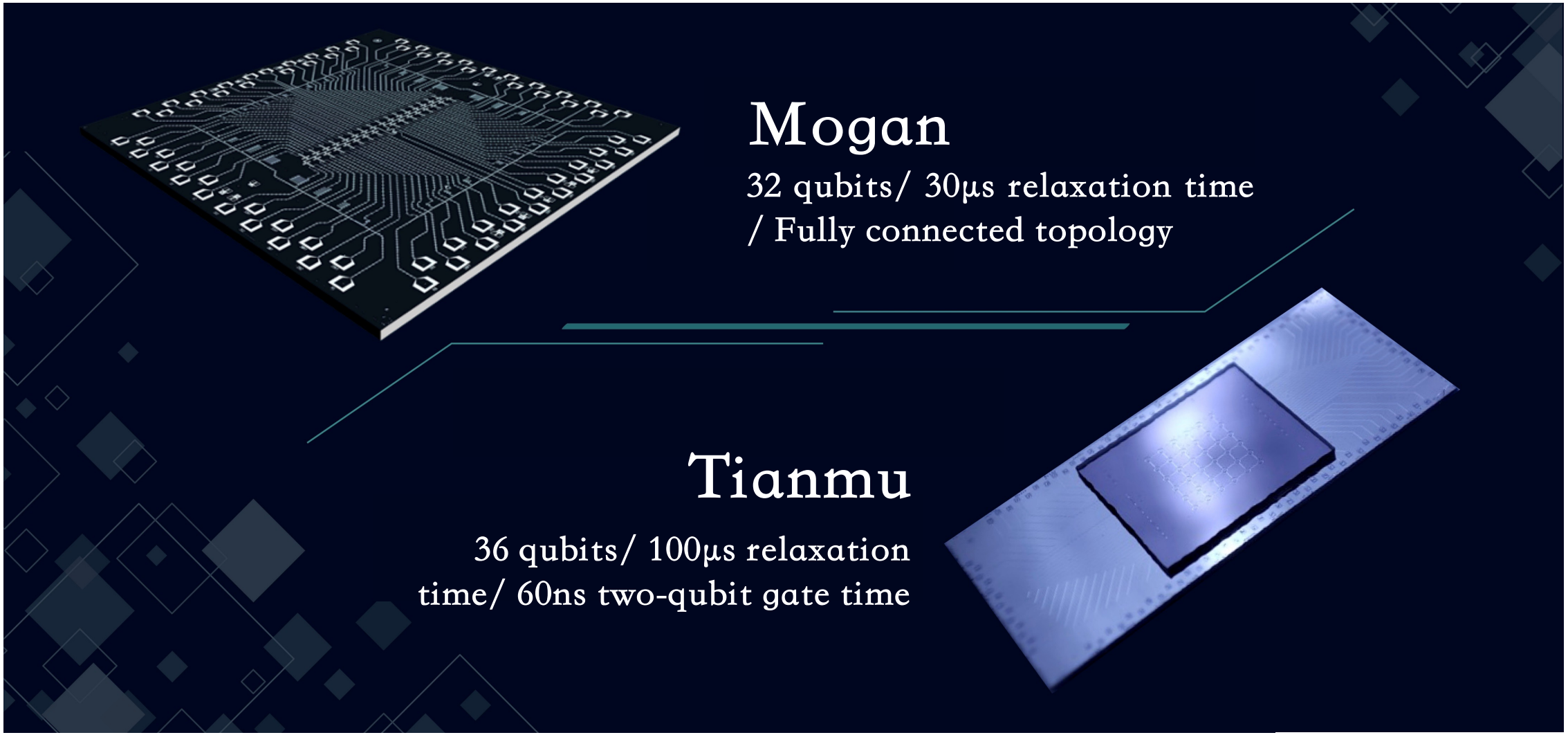


HyQSAT is more suitable to **difficult problem**.

Achieve higher speedup in **larger quantum annealer**.

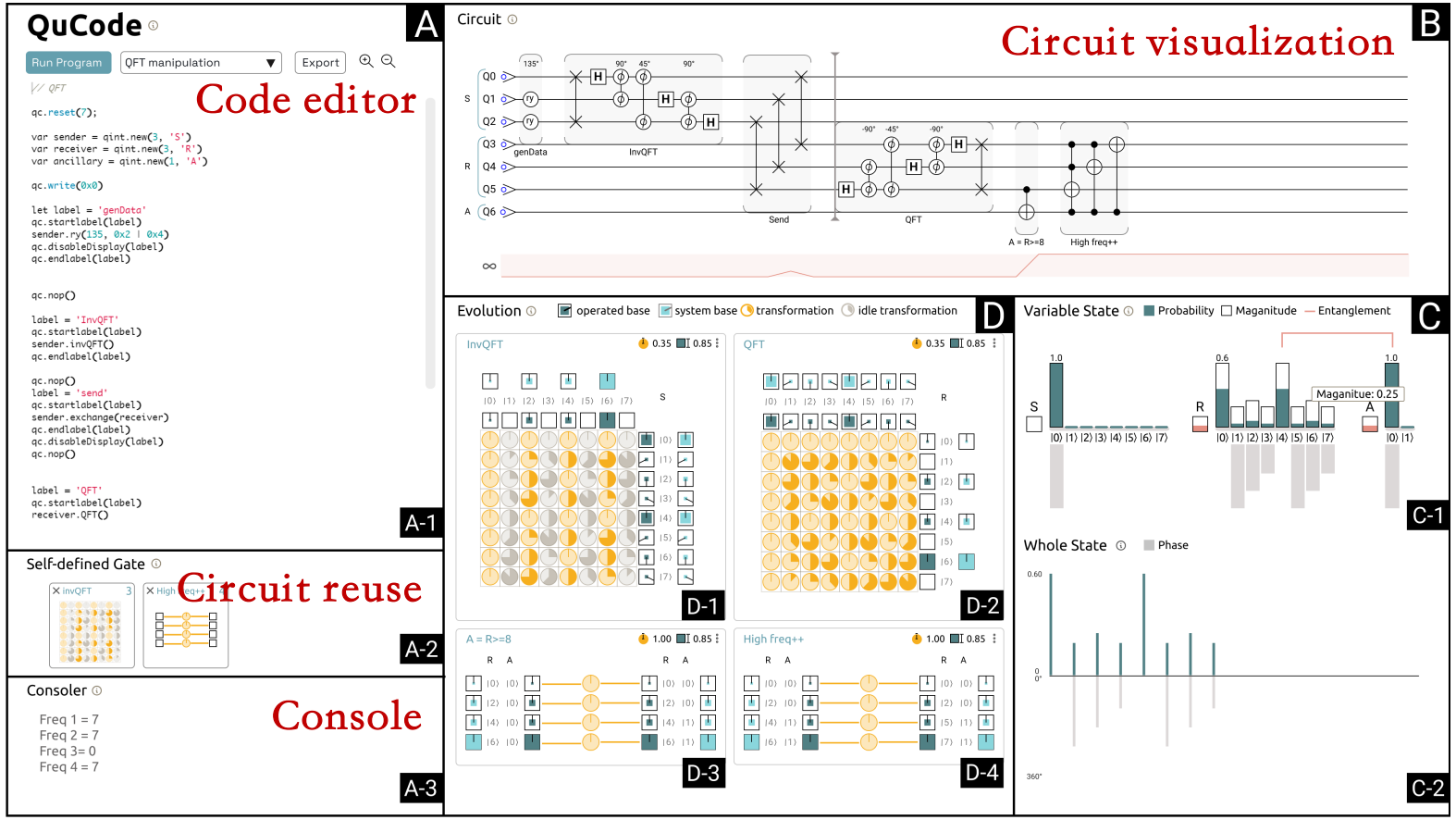
ZHEJIANG UNIVERSITY

By the way: Gate-based Quantum Computer of Zhejiang University

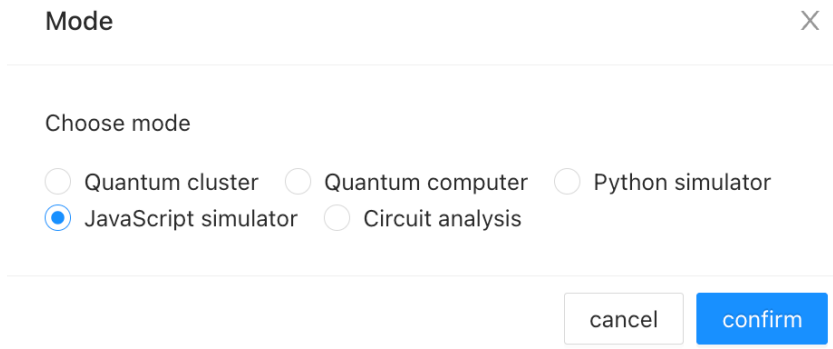


By the way: Our quantum cloud platform (janusq.zju.edu.cn)

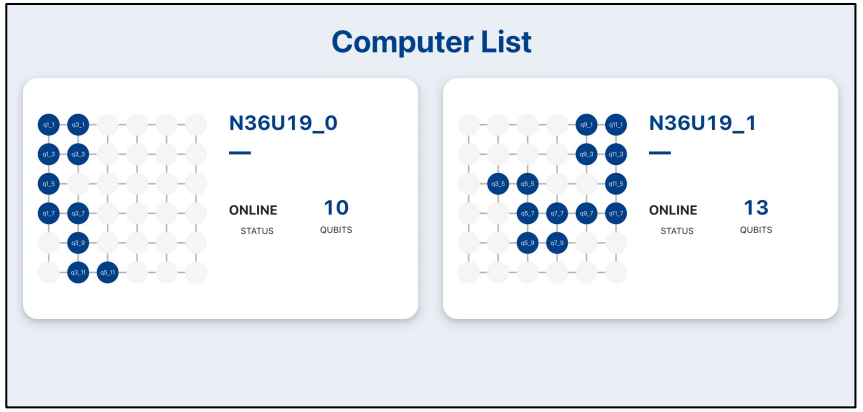
Janus's Visual Programming and Debugging Tools



Multiple execution mode



Connect to two quantum computers



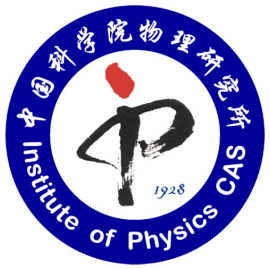
ZHEJIANG UNIVERSITY

太元云平台注册用户

University

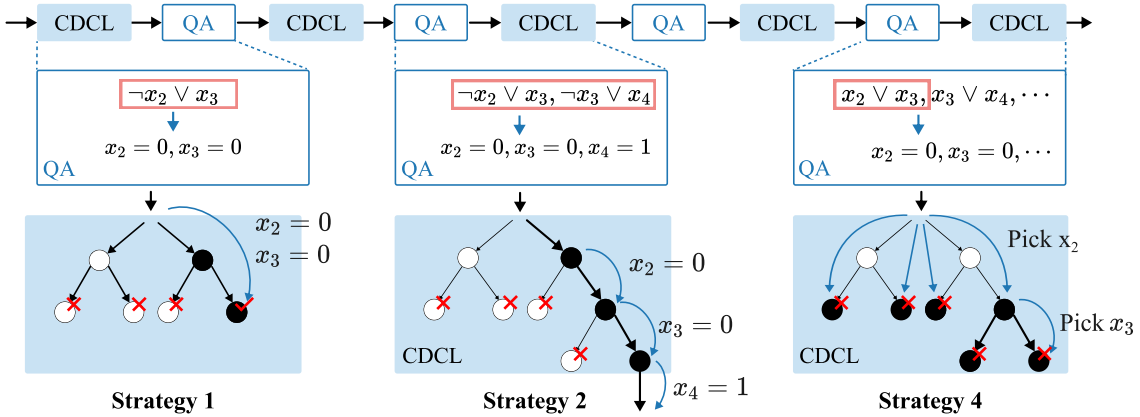
Research institutes

Enterprise

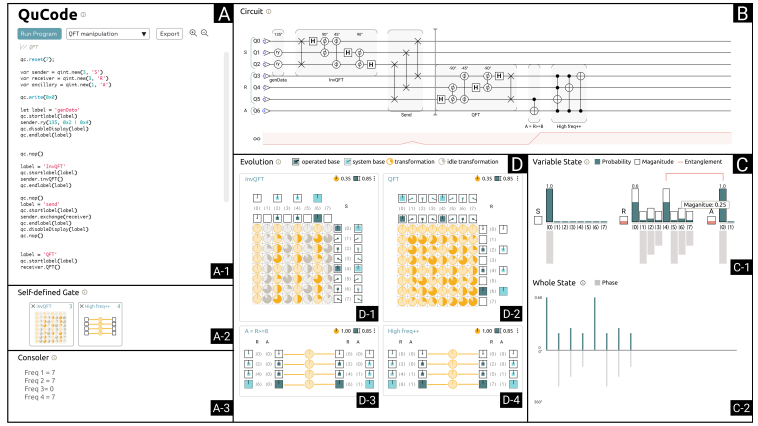


Summary

- Hybrid quantum classical 3-SAT algorithm
  - Locating difficult subproblems
  - Accelerating CDCL search using noisy sub-problem solutions
  - 4.21X speedup compared to the SOTA classical algorithms
- Fast embedding method
  - Avoiding overhead of routing and adjustment
  - Considering locality and qubit reuse
- Janus quantum cloud platform
  - janusq.zju.edu.cn
  - 32 and 36 qubit quantum chips
  - Quantum programming and analysis tools



HyQSAT



JanusQ

ZHEJIANG UNIVERSITY

浙江大学
ZHEJIANG UNIVERSITY

Thanks for listening

HyQSAT: A Hybrid Approach for 3-SAT Problems by Integrating Quantum Annealer with CDCL

Siwei Tan , Mingqian Yu , Andre Python , Yongheng Shang , Tingting Li , Liqiang Lu*, and Jianwei Yin*