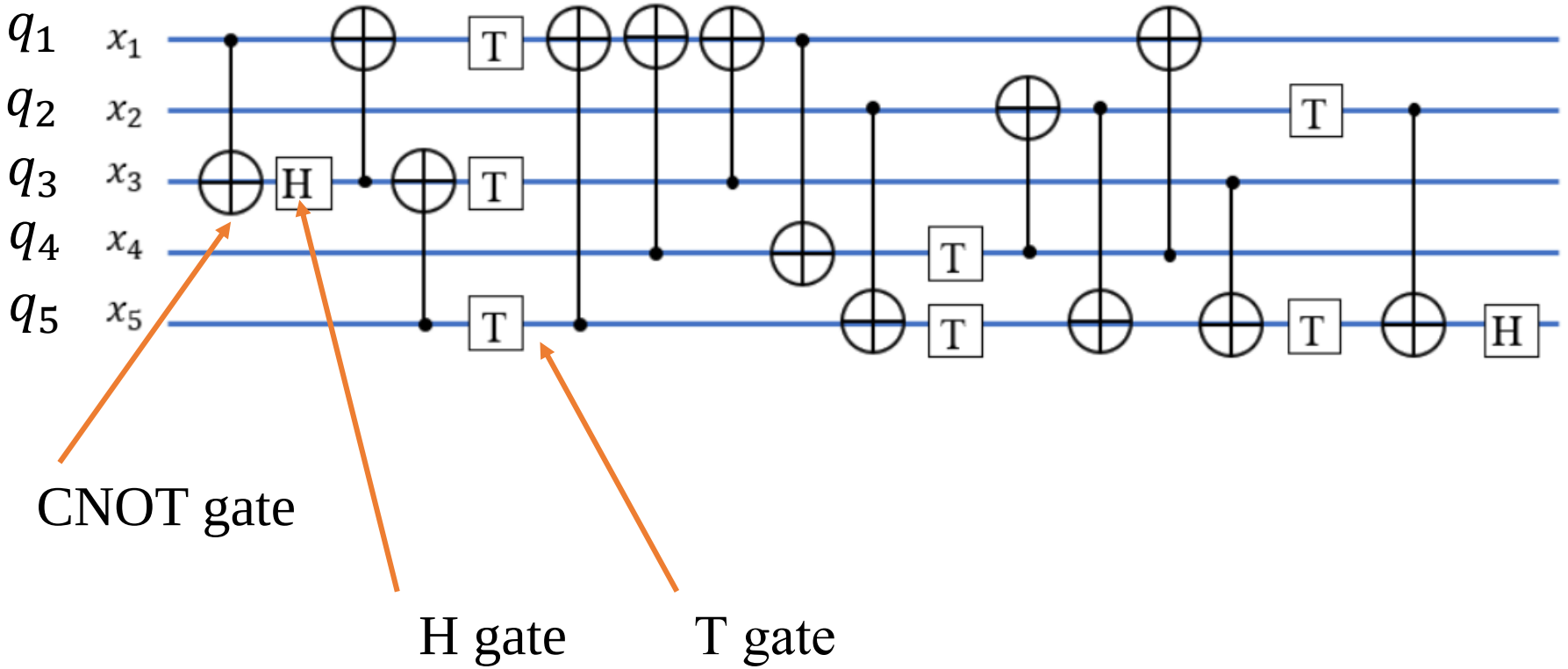
An SMT-Solver-based Synthesis of NNA-Compliant quantum Circuits Consisting of CNOT, H, and T Gate

Kyohei Seino, Shigeru Yamashita

Ritsumeikan University

Quantum Circuits

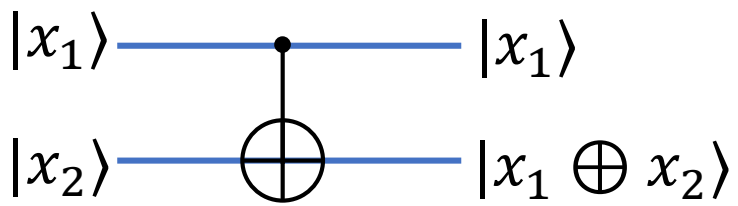
- Diagram of quantum gates in chronological order



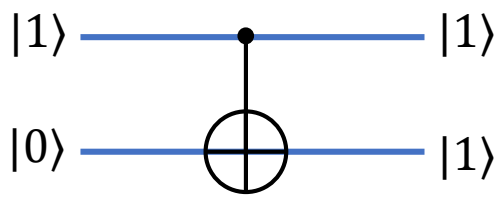
Quantum Gates

A CNOT gate

- Invert x2 when x1=1



- 例

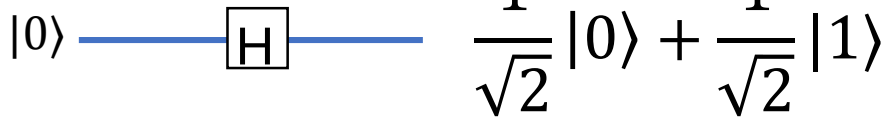


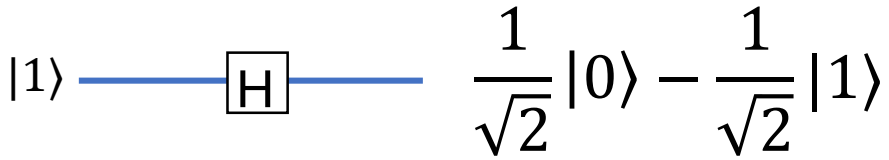
T gate

- Add $e^{i\pi/4}$ phase when x1 =1

H gate

- Generate a supperposed state

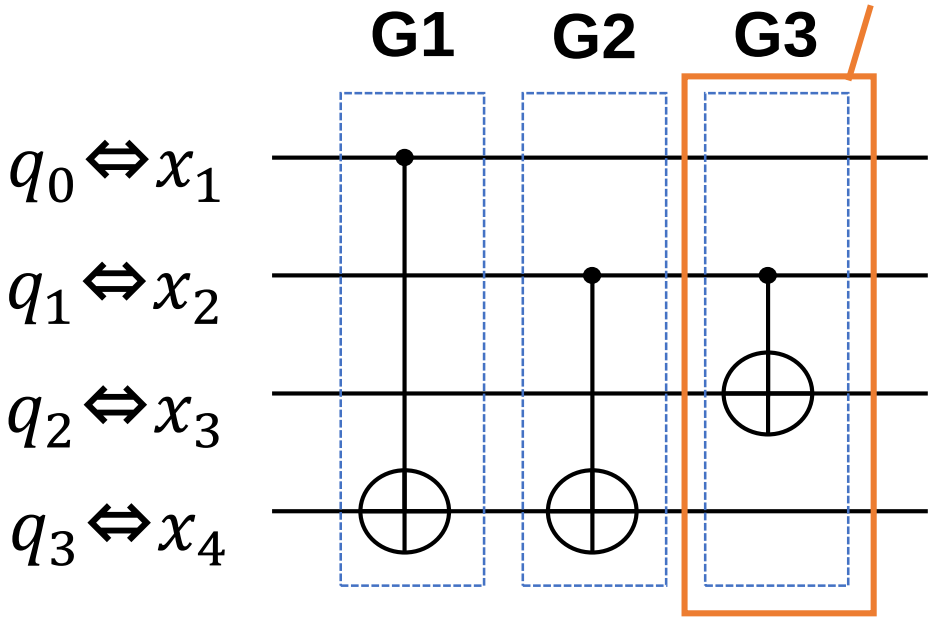


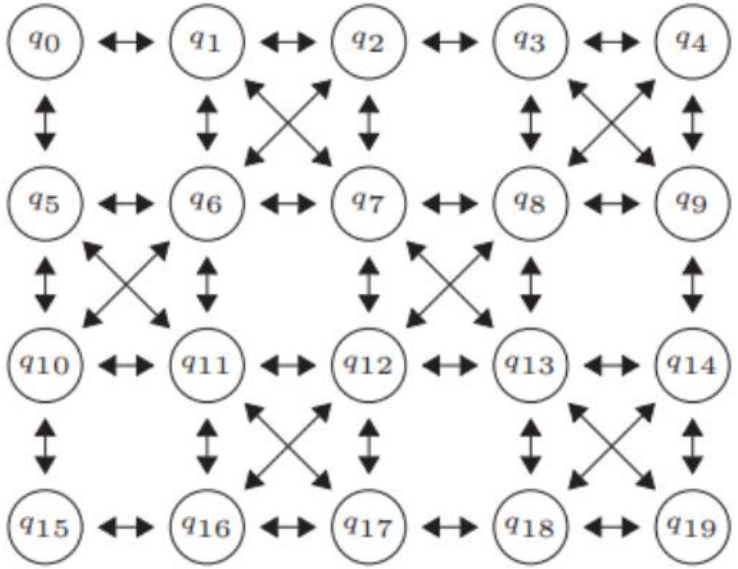


An NNA(Nearest Neighbor Architecture) Restriction

- Permits interactions only between two adjacent physical quantum bits (qubits)

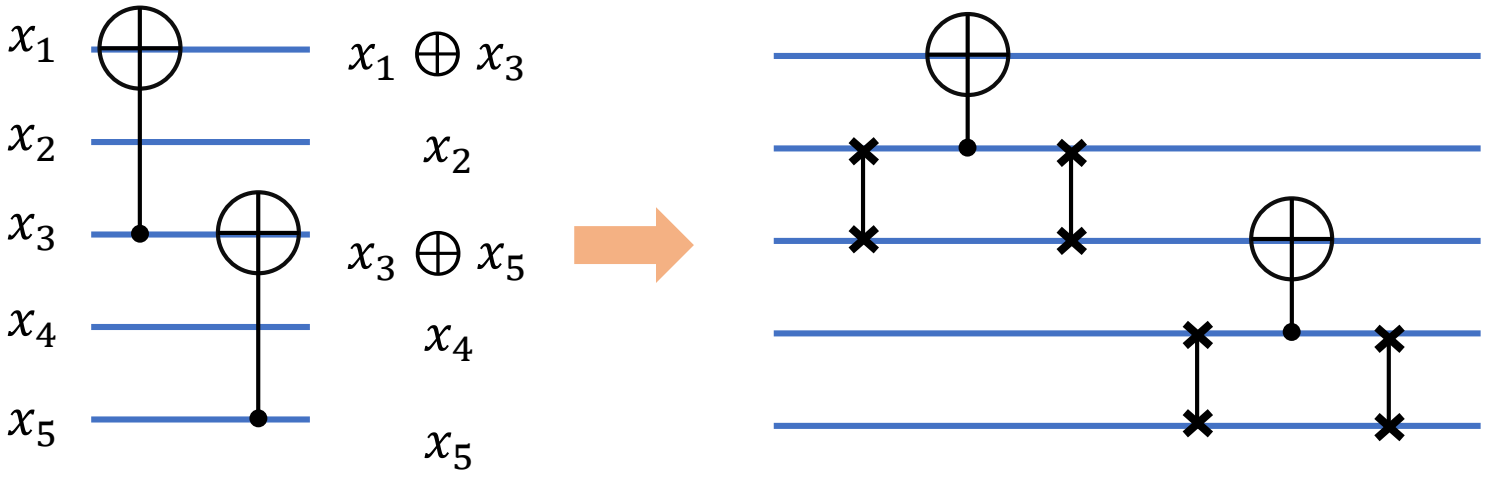
An NNA-Compliant Circuit





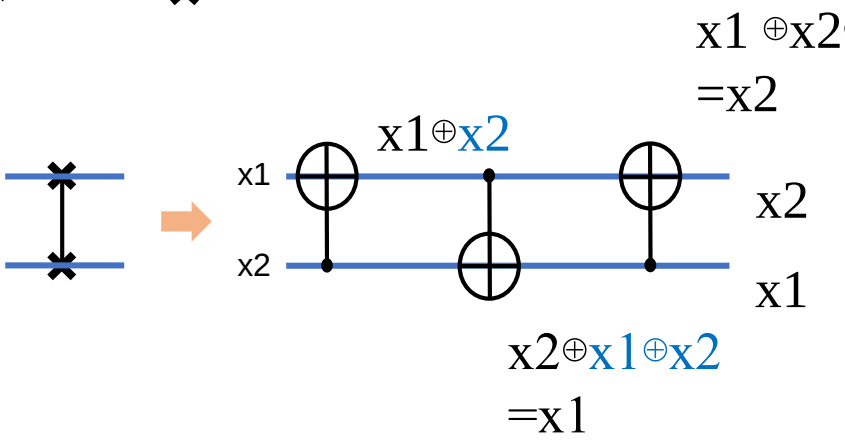
Conversion into NNA-Compliant Circuits

- Conversion with SWAP gate



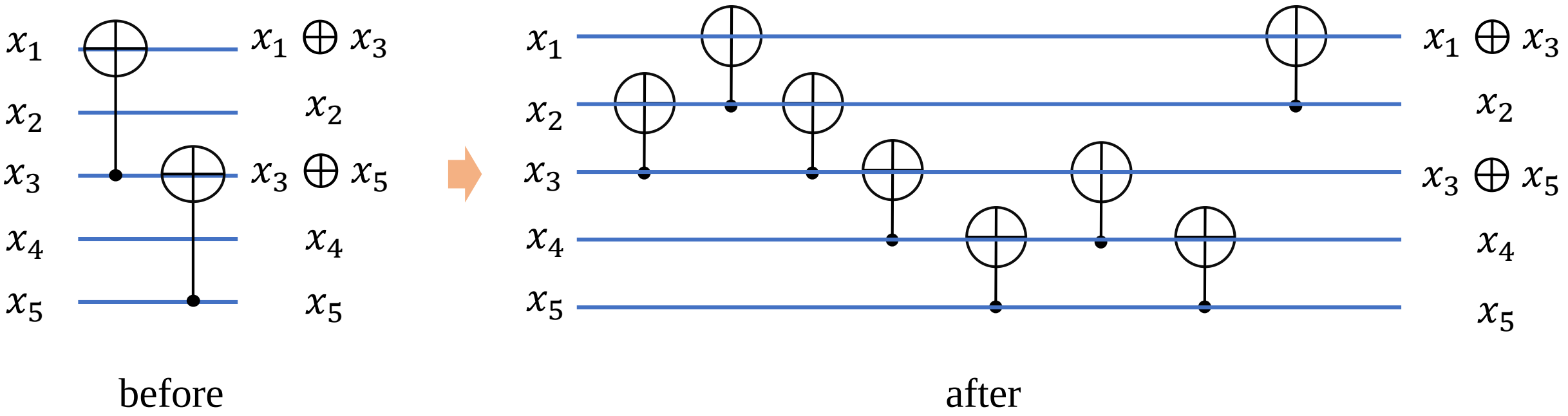
SWAP gate: swap two quantum states

- consist of 3 CNOT gate
- Increase CNOT gate



More efficient Placements

- The circuit like below the diagram without inserting SWAP gate
  - Find circuits consist of NNA-compliant CNOT gates



before

after

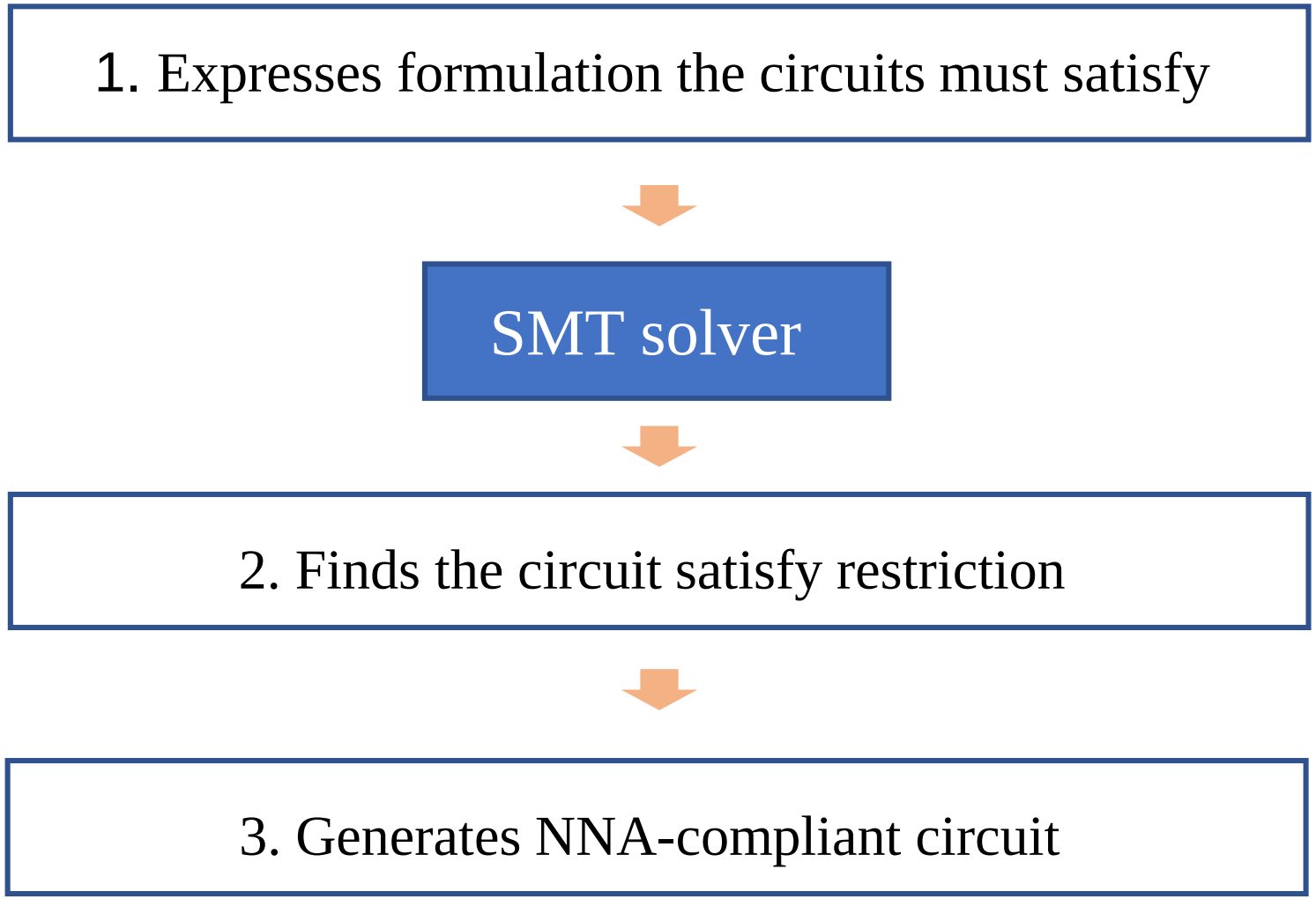
Conversion with SMT Solver

- Construction of NNA-compliant circuits with SMT solver
  - Minimization the number of CNOT gate

- Flow applying SMT solver
  1. Introduces variables
  2. Expresses formulation with variables
  3. Finds a solution with SMT solver

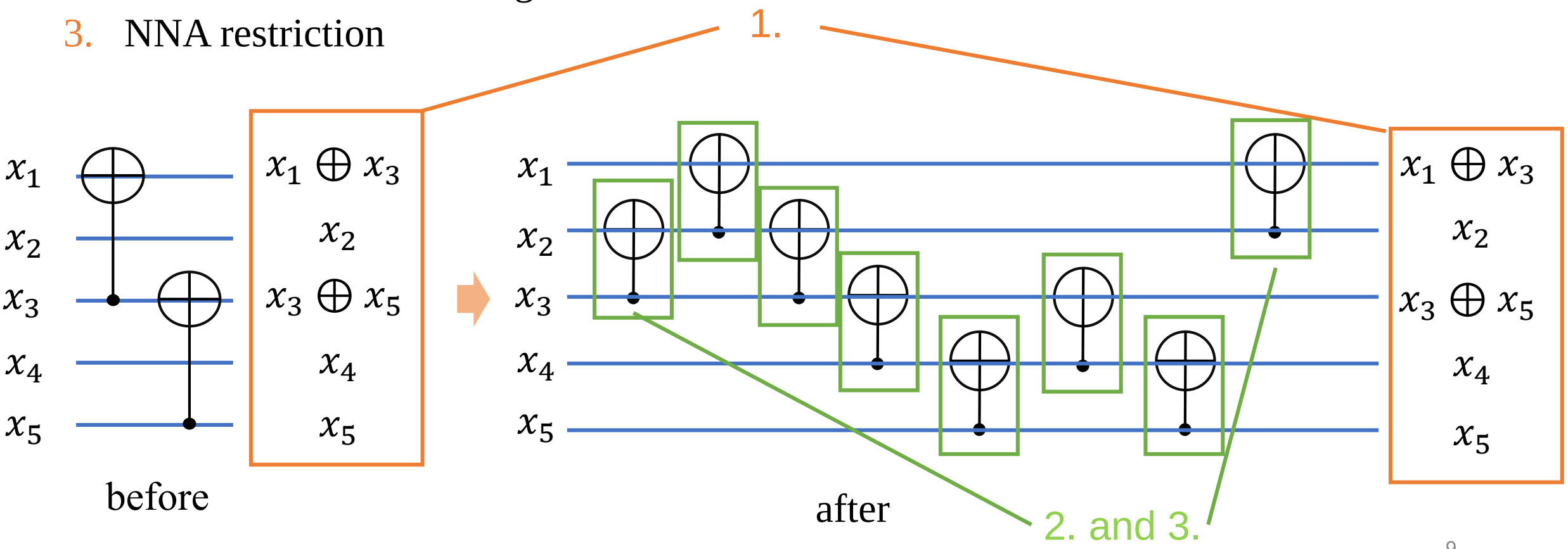
$$(x > 2) \land (0 < y < 3) \land (x + y < 5) \Rightarrow \text{sat}(x=3,y=1)$$

Finding an NNA-compliant Circuit with SMT Solver



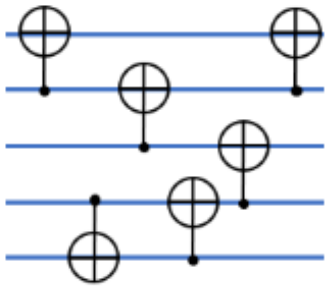
Constraints for an SMT Solver

- Constraints for functional equivalence and the NNA restriction for each CNOT Gate
  1. Functional equivalence
  2. Has a control bit and target bit
  3. NNA restriction



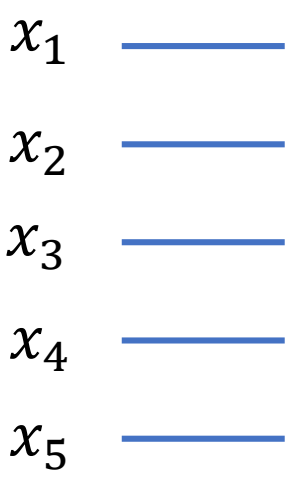
Variables

- CNOT-gate placements

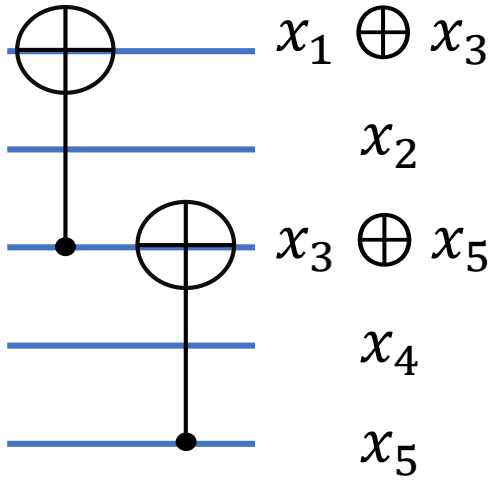


$$Control\ bit\ C_{i,j} = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \quad Target\ bit\ T_{i,j} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

- Inputs

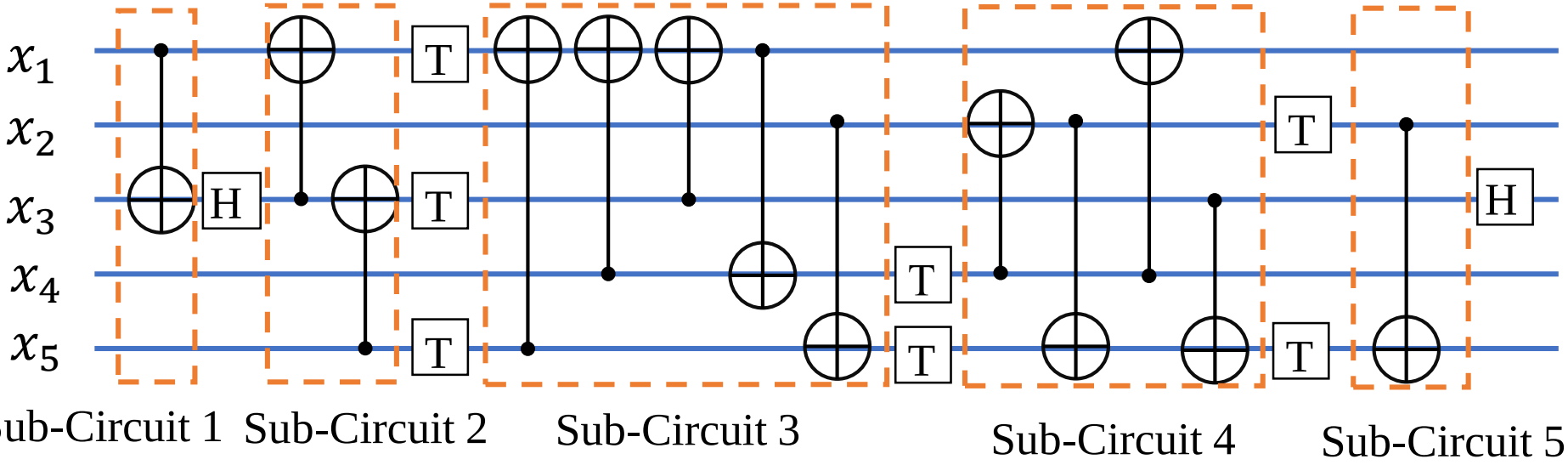


- Logical states
  - Expressed with input variables



Method for Division of The Circuit

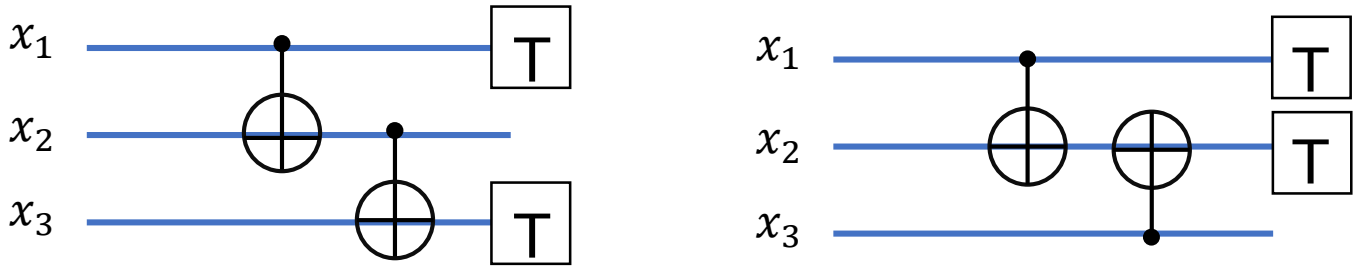
- Divides into subcircuit consisting of CNOT gate
  - Divides the circuit by H/T gates
  - Converts subcircuit into NNA-compliant circuiuts



- Calculations of subcircuits
  - Treats only CNOT gates
  - Linear combination of input variables

Our new observation

- The behaviors of the two T gates are exactly the same
  - two quantum states right before the two T gates are the same

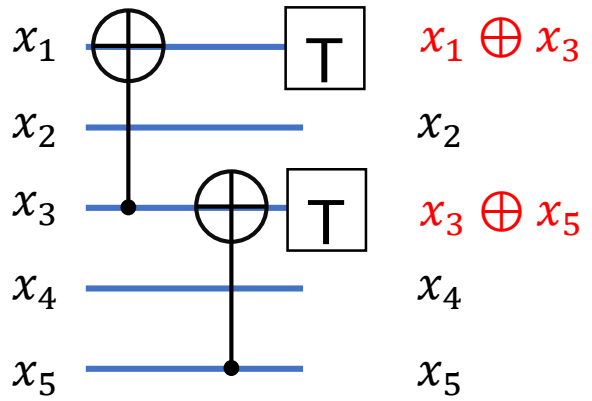


- *Necessary_set*
  - Logical State before T gates
  - Above two circuits have same *necessary_set* = $\{ x_1 \oplus x_2 \oplus x_3, x_1 \}$

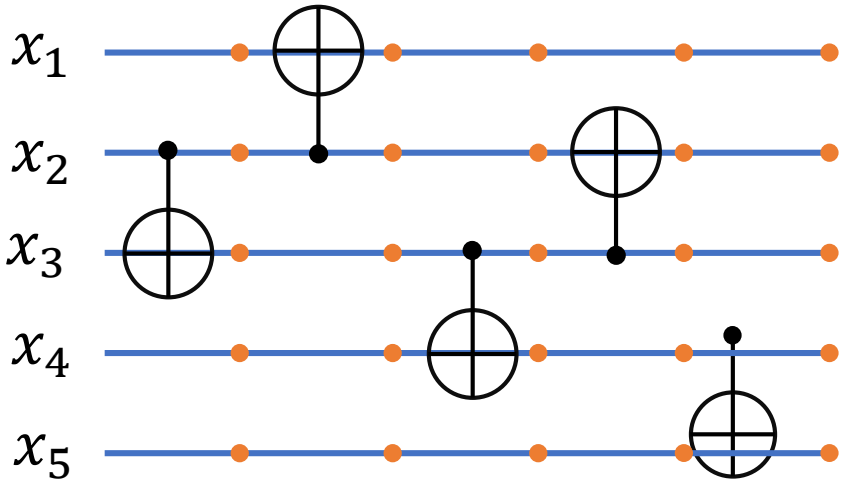
- Functional equivalence
  - Calculates *necessary_set* of *the circuit before conversion*
  - Somewhere in the circuit after conversion, *necessary_set* must be included.

“Don’t care” quantum bit

- Calculates logical states before T gates
  - Only logic states before T gates are considered
  - Logical states in black stand for “Don`t care”

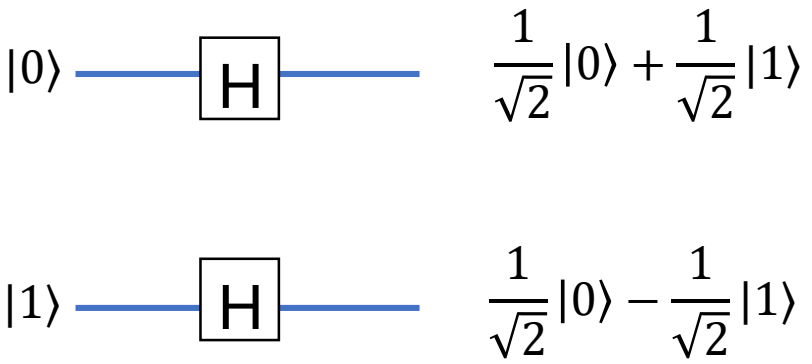


- Converts to satisfy all of these in the circuit after conversion
  - Logic function in red exists somewhere in the circuit
  - *necessary_set*=$\{x_1 \oplus x_3, x_3 \oplus x_5\}$

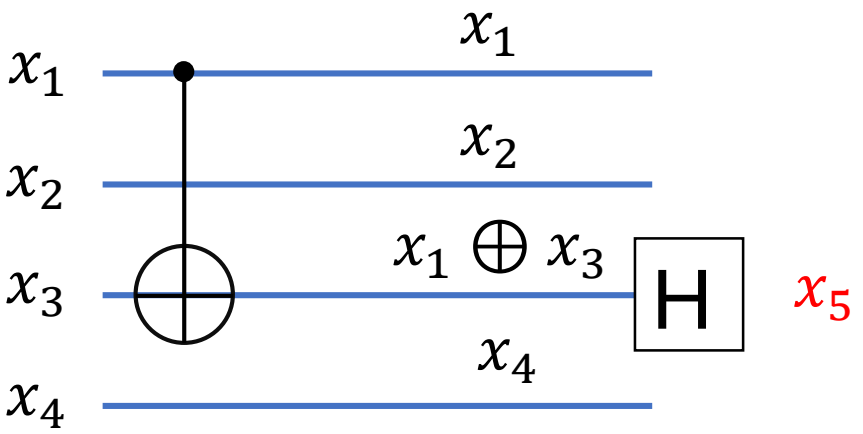


After H gates apply

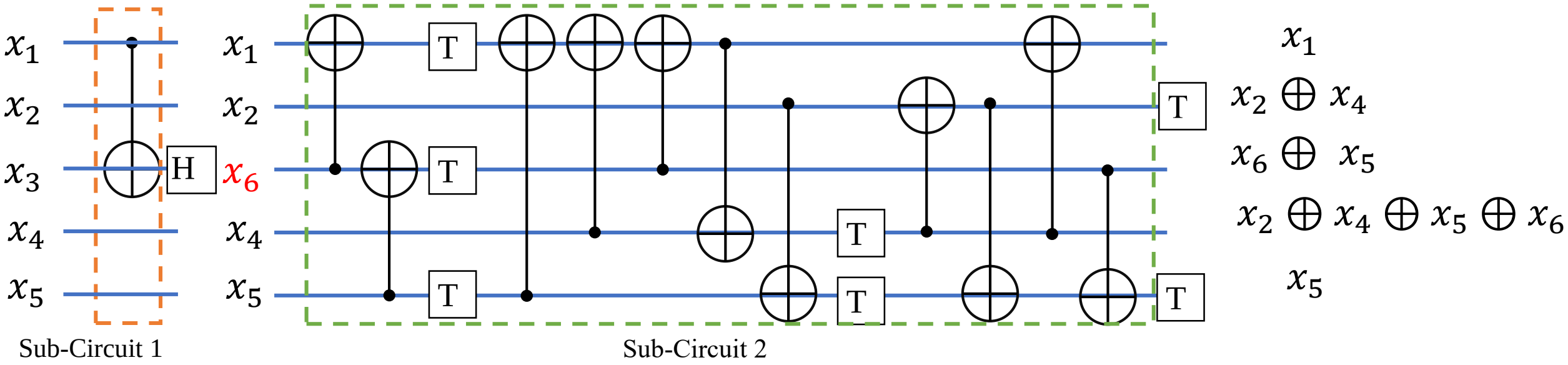
- Quantum states after applying H gates
  - Alter to supperposed states
  - cannot be expressed using the input variables.



- Adds new variables
  - Adds new variables after applying H gates

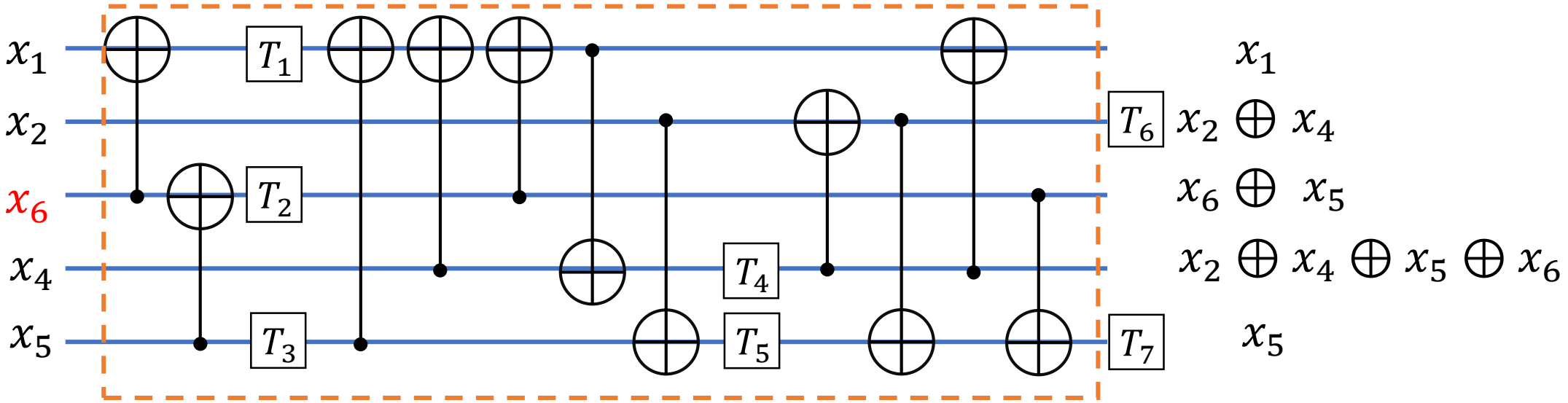


Division of the Circuit in the proposed Method



- Continuous T gates
  - Treats as a subcircuit
  - necessary_set=$\{x_4, x_5, x_6, x_1 \oplus x_6, x_5 \oplus x_6, x_2 \oplus x_4, x_2 \oplus x_4 \oplus x_5\}$
  - In conversion of the circuit in Green, we consider only necessary_set

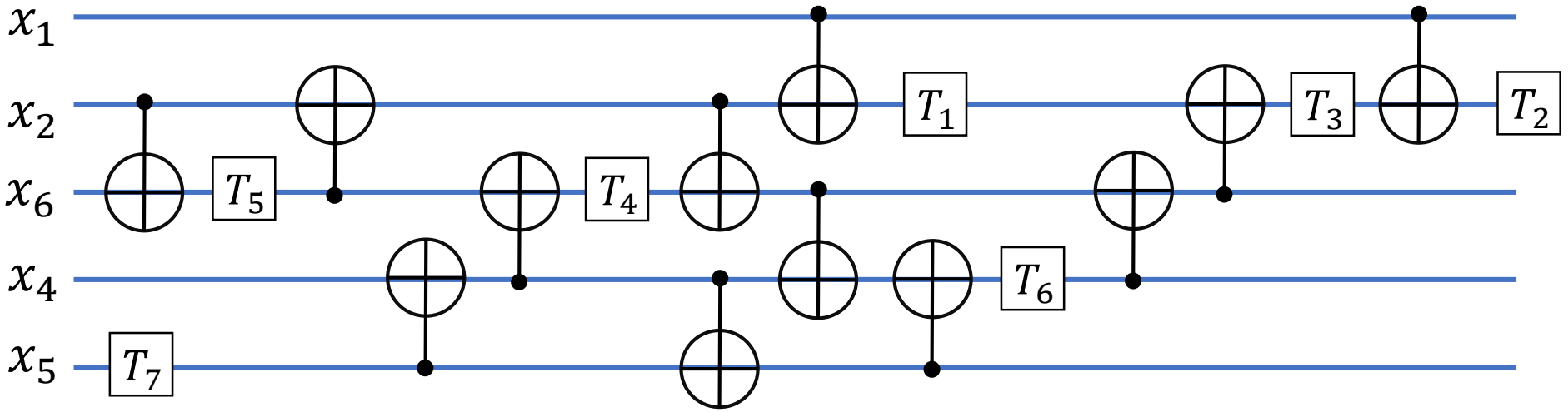
Conversion of the Circuit in the proposed Method



Sub-Circuit 2

- necessary_set = $\{x_1 \oplus x_6 , x_5 \oplus x_6 , x_1 \oplus x_5 \oplus x_6 , x_2 \oplus x_4 \oplus x_5 \oplus x_6 , x_2 \oplus x_5, x_2 \oplus x_4 , x_5 \}$

Results of Conversion



- necessary_set = $\{\boldsymbol{x_1 \oplus x_6}\ ,\ \boldsymbol{x_5 \oplus x_6}\ ,\ \boldsymbol{x_1 \oplus x_5 \oplus x_6}\ ,$
  $\boldsymbol{x_2 \oplus x_4 \oplus x_5 \oplus x_6}\ ,\ \boldsymbol{x_2 \oplus x_5}\ ,\ \boldsymbol{x_5}\ ,\ \boldsymbol{x_2 \oplus x_4}\}$

Conversion into NNA-compliant Circuit with SMT Solver

Benchmark				The number of CNOT gates after transformation			CPU time used by the SMT-solver (seconds)		
Circuit Name	♯ CNOTs	♯ qbits	♯ partition	Ours	[4]	Δ	Ours	[4]	Δ
ham3_102	11	3	3	7	14	-50.0%	0.16	0.22	-16.0%
toffoli_2	14	4	5	19	41	-53.7%	0.79	0.99	-20.3%
alu-v1_28	38	5	11	60	143	-58.0%	6.56	9.27	-29.3%
4gt11_82	18	5	8	23	55	-58.2%	7.82	5.66	38.1%
rd32_271.real	24	5	7	33	93	-64.6%	3.69	13.9	-73.5%
one-two-three-v0_98	65	5	12	85	163	-47.9%	9.35	10.5	-11.1%
fredkin_6.real	21	3	7	17	39	-56.4%	0.16	0.60	-26.8%
mod10_176.real	70	5	12	67	211	-68.2%	4.34	1.59	-36.0%
hwb4_52.real	29	4	9	31	71	-56.3%	0.99	1.59	-37.6%
alu-v2_31.real	108	5	19	270	833	-67.6%	56.4	85.6	-34.1%
aj-e11_168.real	48	4	9	60	148	-59.5%	4.92	15.0	-67.2%
miller_11.real	23	3	7	15	35	-57.1%	0.39	0.54	-26.8%

- Experimental results
  - 29.22% computation time was reduced for the SMT solver
  - 58.11% CNOT gates have been reduced on average

Conclusion

- Conclusion
  - More optimal circuits can be constructed by considering don't-care qubits
  - H gate sub-circuits can only be handled in the same way as existing methods

- Future works
  - Adaptation to larger quantum circuits
  - Conversion without splitting the circuit