AutoQ: Automata-based Quantum Program Analyzer

P. A. Abdulla[2] Y. F. Chen[3] Y.-G. Chen[3] K.-M. Chung[3] M.-H. Hsieh[5] L. Holík[4] W.-J. Huang[5] O. Lengal[4] F.-Y Lo[3] J.-A. Lin[1] W.-L. Tsai[3] D.-D. Yen[6]

[1]iFIRST & CSIE, NTUT [2]Uppsala University [3]IIS, Academia Sinica [4]Brno University of Technology [5]Hon Hai Quantum Computing Research Center [6]MPI SWS

Abstract

We present AutoQ, an analyzer of quantum programs based on a novel symbolic representation of sets of quantum states using *level-level-synchronized tree automata* (LSTAs).

AutoQ verification framework of quantum programs

$$\{\ \overbrace{\mathcal{A}_{Pre}}^{\text{Precondition}}\ \}\ \underbrace{P}_{\text{program}}\ \{\ \overbrace{\mathcal{A}_{Post}}^{\text{Postcondition}}\ \}$$

Meaning:

- $\mathcal{A}_{Pre}$ and $\mathcal{A}_{Post}$ denote **set of quantum states**, encoded as LSTA
- If P is executed from a quantum state from $\mathcal{A}_{Pre}$
- then the quantum state after P terminates in $\mathcal{A}_{Post}$

Verification Algorithm:

1. Start with LSTA $\mathcal{A}_{Pre}$
2. Run P on $\mathcal{A}_{Pre}$ using abstract transformations, obtaining $\mathcal{A}_P$
3. Test $\mathcal{L}(\mathcal{A}_P) \subseteq_{uts} \mathcal{L}(\mathcal{A}_{Post})$

Syntax of quantum programs

AutoQ can handle quantum programs with the following syntax

$$P \ ::= \ U \ \big|\ P;P \ \big|\ \textbf{while } (M_i = b) \textbf{ do } \{P\} \ \big|\ \textbf{if } (M_i = b) \textbf{ then } \{P\} \textbf{ else } \{P\}$$

where P is a quantum program, U is a quantum gate annotated with its control and target qubits (e.g., CX_1^2), $b \in \{0,1\}$, and M_i is the measured value of the i-th qubit.

LSTA as set of quantum states

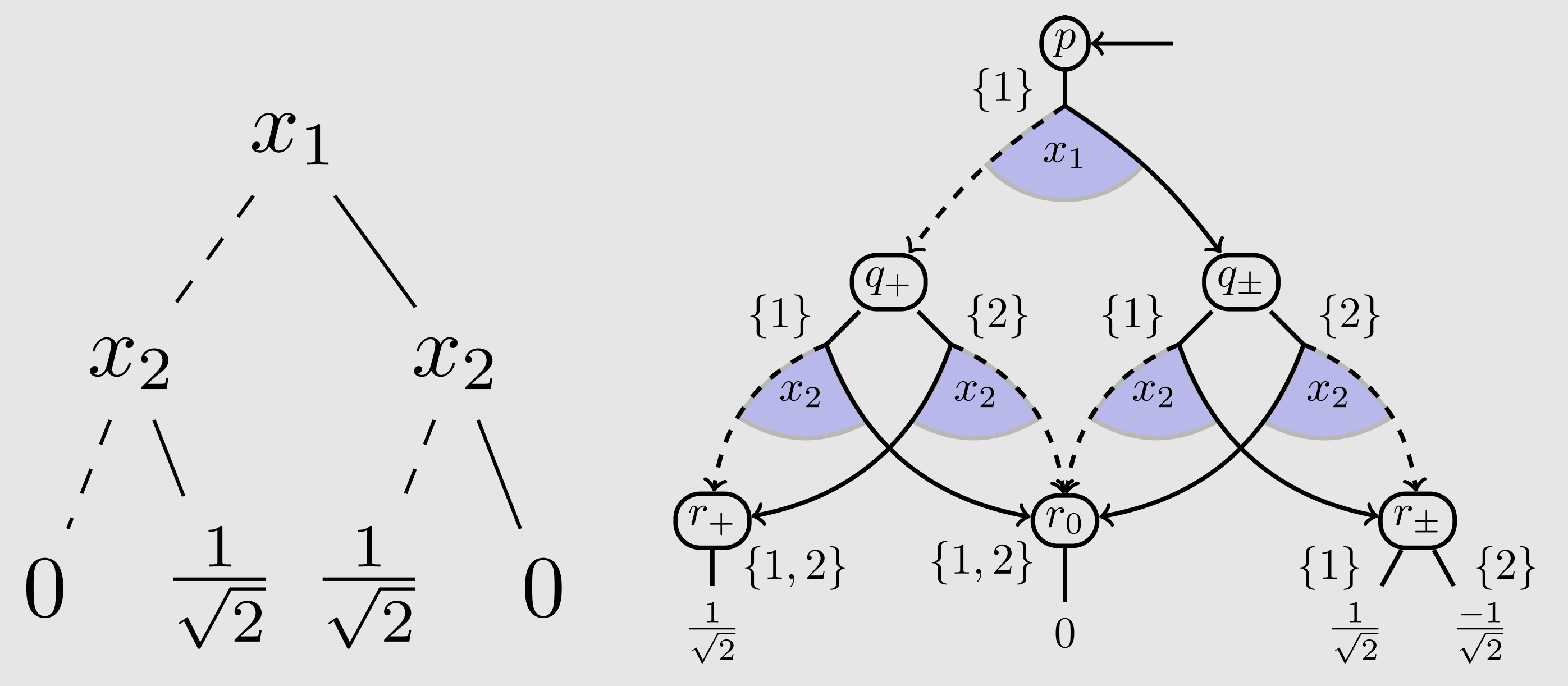


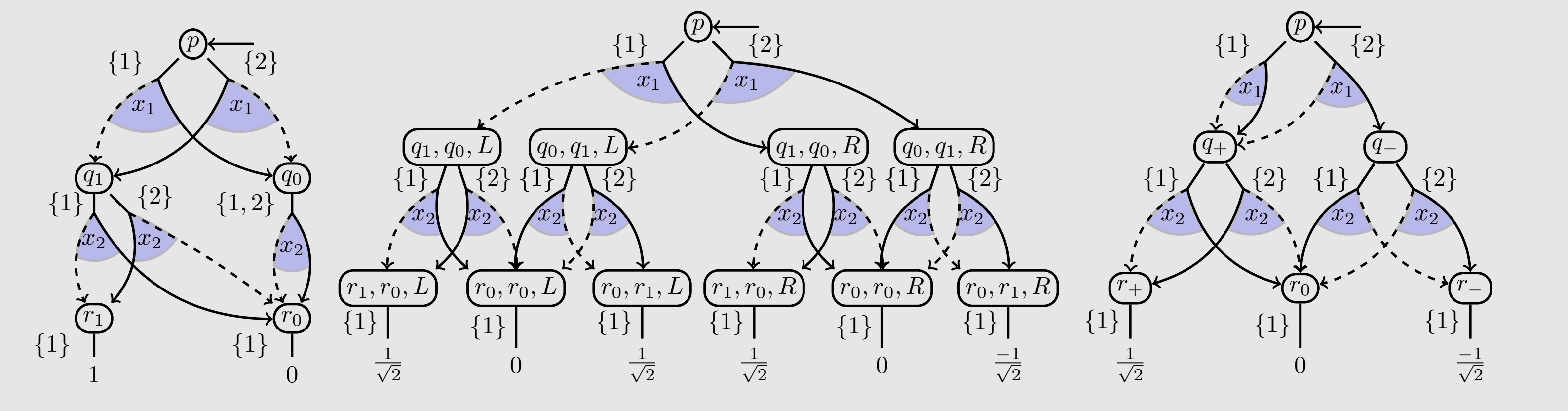
Figure 1. The LSTA for Bell states $\{\frac{1}{\sqrt{2}}(|00\rangle \pm |11\rangle), \frac{1}{\sqrt{2}}(|01\rangle \pm |10\rangle)\}$

By viewing a (pure) quantum state as a binary tree, where each branch represents a *computational basis state*, such as $|10\rangle$ or $|00\rangle$ for a two-qubit circuit. The leaves represent the *complex probability amplitudes* of the state. For example, on the left, the tree represents a state with two qubits where the basis states $|01\rangle$ and $|10\rangle$ have the probability amplitude $\frac{1}{2}$ and the others have amplitude 0.
The set of all 2-qubit Bell states $\{\frac{1}{\sqrt{2}}(|00\rangle \pm |11\rangle), \frac{1}{\sqrt{2}}(|01\rangle \pm |10\rangle)\}$ can be generated by the LSTA on the right. Starting at the root state p, and proceeding iteratively by picking a transition to generate children states, until reaching the leaves. For example, the tree on the left can be generated by first picking the transition $\boxed{p} \xrightarrow{\{1\}} x_1 \boxed{q_+}\boxed{q_\pm}$, then the two transitions, $\boxed{q_+} \xrightarrow{\{2\}} x_2 \boxed{r_0}\boxed{r_+}$, $\boxed{q_\pm} \xrightarrow{\{2\}} x_2 \boxed{r_\pm}\boxed{r_0}$, and ending with the leaf transitions $\boxed{r_+} \xrightarrow{\{1,2\}} \frac{1}{\sqrt{2}}$, $\boxed{r_0} \xrightarrow{\{1,2\}} 0$, $\boxed{r_\pm} \xrightarrow{\{1\}} \frac{1}{\sqrt{2}}$.

Properties:

- Allow synchronization across subtrees
- The inclusion of languages of two LSTAs is decidable
- Incomparable to basic tree automata

Quantum operations as LSTA transformations



(a) Basis states (b) Applied H_1 to the LSTA from left (c) Reduced LSTA

Figure 2. An example of applying a single-qubit gate on a set of quantum states represented using an LSTA

With such a representation of quantum states, a quantum gate U is then encoded as tree transformations. For example, applying a 1-qubit gate $U = \left(\begin{smallmatrix} u_1 & u_2 \\ u_3 & u_4 \end{smallmatrix}\right)$ to the t-th qubit of a quantum state combines the 0-subtree T_0 and the 1-subtree T_1 under each node labelled with x_t. For every pair of T_0 and T_1 under a node labeled x_t, with leaf amplitudes of $a_1, a_2, \ldots, a_k$ and $b_1, b_2, \ldots, b_k$ respectively, the new state will have a new 0-subtree with the amplitudes $(u_1 \cdot a_1 + u_2 \cdot b_1), (u_1 \cdot a_2 + u_2 \cdot b_2), \ldots, (u_1 \cdot a_k + u_2 \cdot b_k)$, and a 1-subtree with the amplitudes $(u_3 \cdot a_1 + u_4 \cdot b_1), (u_3 \cdot a_2 + u_4 \cdot b_2), \ldots, (u_3 \cdot a_k + u_4 \cdot b_k)$.
The construction is lifted from trees to LSTAs. An example of applying H gate to an LSTA is shown above, and the result is obtained by applying the LSTA reduction algorithm to simplify its structure further.

Cost of operations: from $O(|\mathcal{A}|)$ to $O(|\mathcal{A}|^2)$

Example of verifying a quantum program

Algorithm 1: A Weakly Measured Version of Grover's algorithm (solution $s = 0^n$)

```
Pre: $\{1\,|0^{n+2}\rangle + 0\,|*\rangle\}$;
$H_3;\ H_4;\ \ldots;\ H_{n+2}$;
$O_{2,\ldots,(n+2)}; CK_1^2; O_{2,\ldots,(n+2)}$;
Inv: $\{v_{sol1}\,|000^n\rangle + v_k\,|000^{n-1}1\rangle + \cdots + v_k\,|001^n\rangle + v_{sol2}\,|100^n\rangle + 0\,|*\rangle\}$;
while $M_1 = 0$ do
    $\{\mathcal{G}_{2,\ldots,(n+2)}; O_{2,\ldots,(n+2)}; CK_1^2; O_{2,\ldots,(n+2)}\}$;
Post: $\{1\,|10s\rangle + 0\,|*\rangle\}$;
```

A variation of Grover's search, called the *weakly measured* version, was recently proposed. The weakly measured version eliminates the need for knowing the number of solutions, making the algorithm more applicable.
The results of the verification of weakly measured Grover's search are in the left-hand side of below: AutoQ was able to verify the program w.r.t. the specification even for larger numbers of qubits in reasonable time.

Weakly Measured Grover's Search						*Repeat-Until-Success*					
program	**qubits**	**gates**	**result**	**time**	**memory**	**program**	**qubits**	**gates**	**result**	**time**	**memory**
WMGrover (03)	7	50	OK	0.0s	42MB	$(2X + \sqrt{2}Y + Z)/\sqrt{7}$	2	29	OK	0.0s	7MB
WMGrover (10)	21	169	OK	0.2s	42MB	$(I + i\sqrt{2}X)/\sqrt{3}$	2	17	OK	0.0s	7MB
WMGrover (20)	41	339	OK	0.8s	42MB	$(I + 2iZ)/\sqrt{5}$	2	27	OK	0.0s	6MB
WMGrover (30)	61	509	OK	2.3s	43MB	$(3I + 2iZ)/\sqrt{13}$	2	43	OK	0.0s	7MB
WMGrover (40)	81	679	OK	5.4s	43MB	$(4I + iZ)/\sqrt{17}$	2	77	OK	0.0s	6MB
WMGrover (50)	101	849	OK	11s	44MB	$(5I + 2iZ)/\sqrt{29}$	2	69	OK	0.0s	7MB

Application to circuit equivalence checking

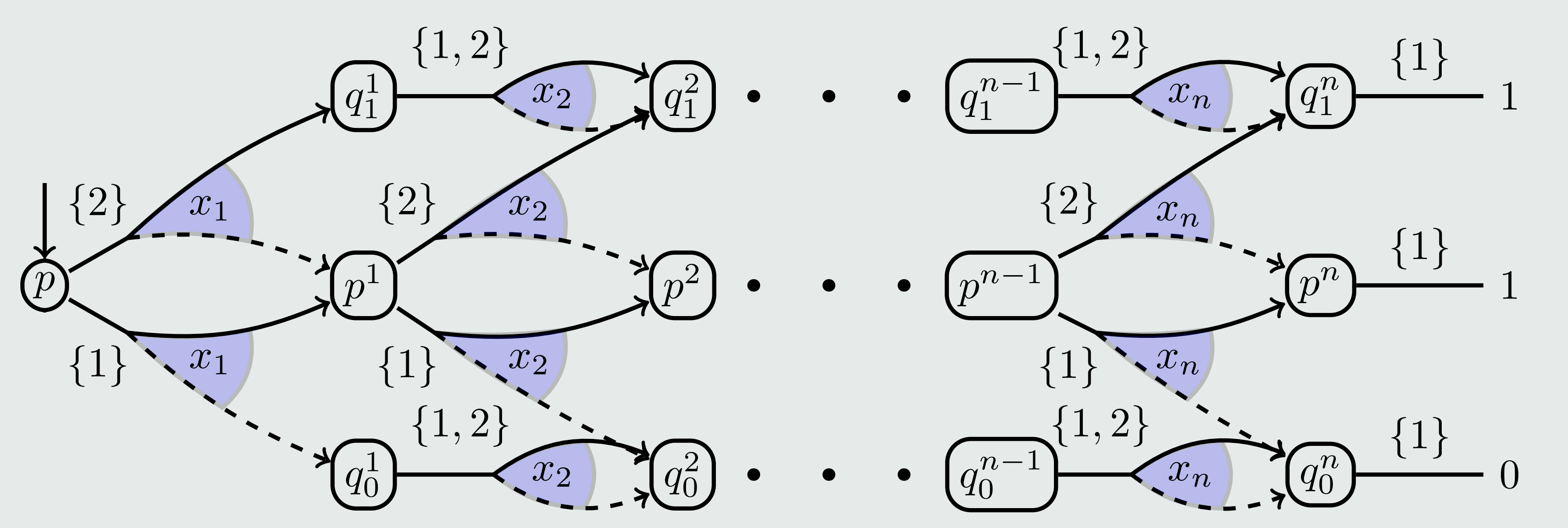
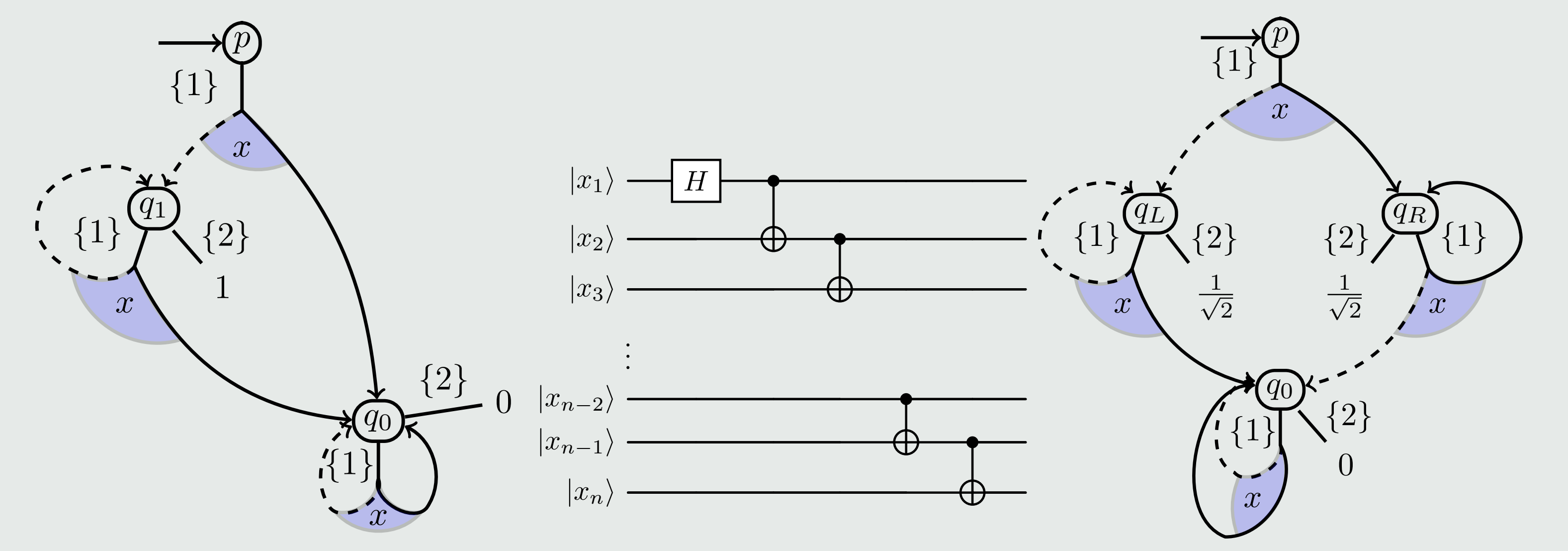


Figure 3. LSTA for the pre- and post-condition for equivalence checking

Given two quantum circuits C_1 and C_2, one can check if they are equivalent by checking if $C_1 C_2^\dagger$ is an identity, where $C_2^\dagger$ can be obtained from C_2 by reverting it and substituting every gate by its inverse. We can then verify if an n-qubit circuit is an identity by testing it against 2^n linearly independent *vectors* and checking if each resulting vector matches its input vector. Using LSTAs, all 2^n of these tests can be performed simultaneously. We use trees corresponding to the set of vectors $\{(0, 0, \ldots, 1), \ldots, (0, 1, \ldots, 1), (1, 1, \ldots, 1)\}$ simultaneously as both the precondition and the postcondition.

Application to parameterized verification



LSTAs enables automated verification of a certain class of parameterized circuits. An example can be found in the above figure. In the middle is the n-qubit GHZ circuit that transforms a quantum state of the form $|0^n\rangle$ to the GHZ state $\frac{|0^n\rangle + |1^n\rangle}{\sqrt{2}}$, for each $n \geq 1$. AutoQ supports several *parameterized gates* that implement sequences of quantum gates applied in a given pattern.

		AutoQ		
	#G	`post`$_C$	$\subseteq$	total
GHZ	2	0.0s	0.0s	**0.0s**
DHS	4	0.0s	0.0s	**0.0s**
single fermionic	14	0.3s	0.0s	**0.3s**
double fermionic	88	1m54s	0.0s	**1m54s**

References

- Yu-Fang Chen, Kai-Min Chung, Min-Hsiu Hsieh, Wei-Jia Huang, Ondřej Lengál, Jyun-Ao Lin, Wei-Lun Tsai, *AutoQ 2.0: From Verification of Quantum Circuits to Verification of Quantum Programs*, In: Gurfinkel, A., Heule, M. (eds) Tools and Algorithms for the Construction and Analysis of Systems. TACAS 2025. Lecture Notes in Computer Science, vol 15698. Springer, Cham.
- Parosh Aziz Abdulla, Yo-Ga Chen, Yu-Fang Chen, Lukáš Holík, Ondřej Lengál, Fang-Yi Lo Jyun-Ao Lin, and Wei-Lun Tsai. 2024. *Verifying Quantum Circuits with Level-Synchronized Tree Automata.* Proc. of the ACM on Program. Lang. 9, POPL (2025), 923-953
- Yu-Fang Chen, Kai-Min Chung, Ondrej Lengál, Jyun-Ao Lin, and Wei-Lun Tsai. 2023. *AutoQ: An Automata-Based Quantum Circuit Verifier.* In Computer Aided Verification - 35th International Conference, CAV 2023, Paris, France, July 17-22, 2023, Proceedings, Part III (Lecture Notes in Computer Science, Vol. 13966), Constantin Enea and Akash Lal (Eds.). Springer, 139–153.
- Yu-Fang Chen, Kai-Min Chung, Ondrej Lengál, Jyun-Ao Lin, Wei-Lun Tsai, and Di-De Yen. *An Automata-Based Framework for Verification and Bug Hunting in Quantum Circuits.* Proc. ACM Program. Lang. 7, PLDI (2023), 1218–1243
- AutoQ `https://github.com/fmlab-iis/AutoQ/`