Automatic Symbolic Analysis of Quantum Programs

Christian Schilling
christianms@cs.aau.dk

August 23, 2023

Acknowledgments

Joint work[1] with Fabian Bauer-Marquart and Stefan Leue

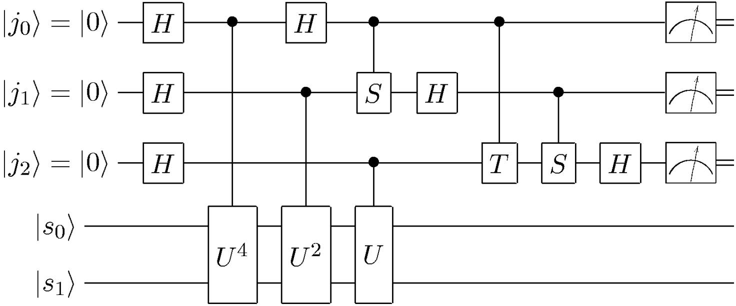
[1] F. Bauer-Marquart, S. Leue, and C. Schilling. *Formal Methods.* 2023.

Overview

Overview

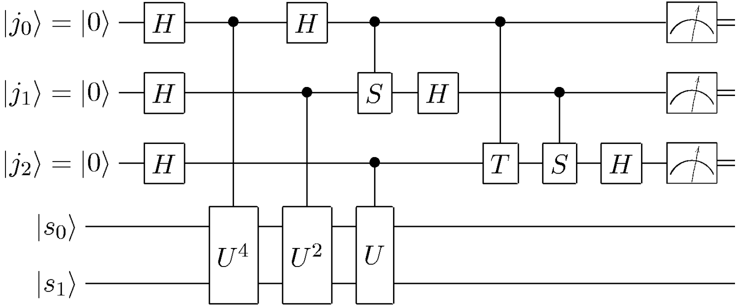
Motivation

- Some interesting problems for quantum programs
  - Correctness analysis
  - Program equivalence
  - Repair of errors and program synthesis
  - Optimization (e.g., number of gates, types of gates, physical implementation adhering to constraints)



Motivation

- Some interesting problems for quantum programs
  - **Correctness analysis** ($\leftarrow$ focus in this presentation)
  - **Program equivalence** ($\leftarrow$ focus in this presentation)
  - Repair of errors and program synthesis
  - Optimization (e.g., number of gates, types of gates, physical implementation adhering to constraints)



Correctness analysis for classical programs

- Verification of classical programs is well studied
    - Given are a program and a specification
    - Decide whether specification holds

```
int f91(int x) {
   if (x > 100)
      return x - 10;
   else
      return f91(f91(x + 11));
}
int main() {
   int x = user_input();
   int res = f91(x);
   assert (x > 100 && res == x - 10) || res == 91
}
```

$$f91(x) = \begin{cases} x - 10 & x > 100 \\ 91 & \text{otherwise} \end{cases}$$

Correctness analysis for quantum programs

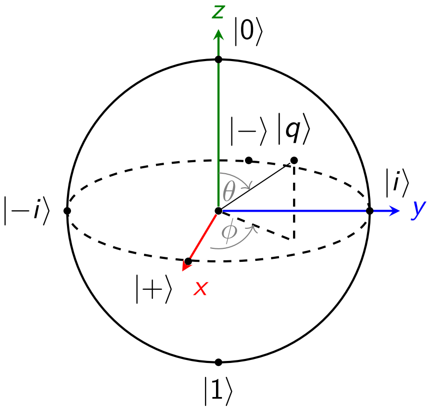
- This presentation
  - Verification against specifications in first-order logic
  - Reduction to an automatic solution technique
  - Efficient encoding and overapproximation

Overview

Qubit

- Ground state $|0\rangle$
- Excited state $|1\rangle$
- Superposition $|q\rangle = \alpha\,|0\rangle + \beta\,|1\rangle, \quad \alpha, \beta \in \mathbb{C}$
- Written as 2D vector: $|q\rangle \equiv \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$
- Constraint $|\alpha|^2 + |\beta|^2 = 1$

Bloch sphere



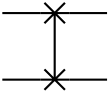
- Polar coordinates: $|q\rangle = \underbrace{\cos\frac{\theta}{2}}_{\alpha}|0\rangle + \underbrace{e^{i\phi}\sin\frac{\theta}{2}}_{\beta}|1\rangle$

Multiple qubits (unentangled)

$$
\begin{aligned}
|q_0, q_1\rangle &= |q_0\rangle \otimes |q_1\rangle \equiv \begin{bmatrix} \alpha_0 \\ \beta_0 \end{bmatrix} \otimes \begin{bmatrix} \alpha_1 \\ \beta_1 \end{bmatrix} \\
&= \alpha_0\alpha_1 |00\rangle + \alpha_0\beta_1 |01\rangle + \beta_0\alpha_1 |10\rangle + \beta_0\beta_1 |11\rangle \\
&\equiv \begin{bmatrix} \alpha_0\alpha_1 \\ \alpha_0\beta_1 \\ \beta_0\alpha_1 \\ \beta_0\beta_1 \end{bmatrix}
\end{aligned}
$$

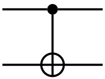
Quantum gates

- Unitary matrix operations
- Example: swapping of two qubits



$$SWAP(|q_0, q_1\rangle) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \alpha_0\alpha_1 \\ \alpha_0\beta_1 \\ \beta_0\alpha_1 \\ \beta_0\beta_1 \end{bmatrix} = \begin{bmatrix} \alpha_0\alpha_1 \\ \beta_0\alpha_1 \\ \alpha_0\beta_1 \\ \beta_0\beta_1 \end{bmatrix} = |q_1, q_0\rangle$$

Another quantum gate: Controlled NOT



$$CNOT(|q_0, q_1\rangle) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \alpha_0\alpha_1 \\ \alpha_0\beta_1 \\ \beta_0\alpha_1 \\ \beta_0\beta_1 \end{bmatrix} = \begin{bmatrix} \alpha_0\alpha_1 \\ \alpha_0\beta_1 \\ \beta_0\beta_1 \\ \beta_0\alpha_1 \end{bmatrix}$$

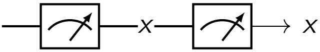
$$= |q_0, q_0 \oplus q_1\rangle$$

where $\oplus$ is addition modulo 2 ("XOR")

Measurement

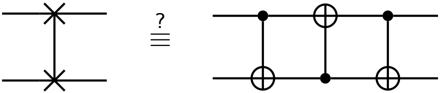
- "Converts" to basis state ($|0\rangle$ or $|1\rangle$)
- Not invertible

$$|q\rangle \equiv \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$

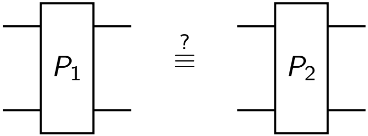


$$x = \begin{cases} |0\rangle & \text{with probability } |\alpha|^2 \\ |1\rangle & \text{with probability } |\beta|^2 \end{cases}$$

Are these two quantum programs equivalent?

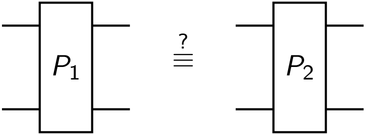


Black-box equivalence check



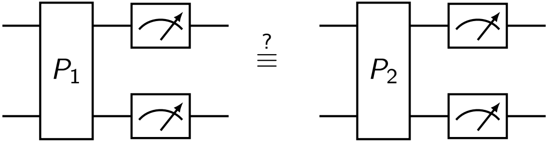
- Idea: pick an initial state and execute both programs
  - One-way: disagreement implies different programs

Black-box equivalence check



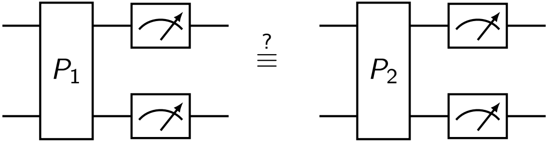
- Idea: pick an initial state and execute both programs
  - One-way: disagreement implies different programs
- **Cannot obtain exact quantum state**

Black-box equivalence check



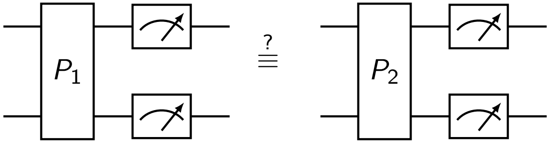
- Idea: pick an initial state and execute both programs
  - One-way: disagreement implies different programs
- **Cannot obtain exact quantum state**
  - Measurement only yields a basis state

Black-box equivalence check



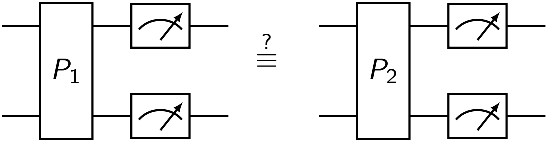
- Idea: pick an initial state and execute both programs
  - One-way: disagreement implies different programs
- **Cannot obtain exact quantum state**
  - Measurement only yields a basis state
  - Disagreement does **not** imply different programs

Black-box equivalence check



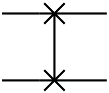
- Idea: pick an initial state and execute both programs
  - One-way: disagreement implies different programs
- **Cannot obtain exact quantum state**
  - Measurement only yields a basis state
  - Disagreement does **not** imply different programs
  - Approximation by executing many times
    - **Expensive and no guarantee**

Black-box equivalence check

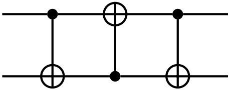


- Idea: pick an initial state and execute both programs
  - One-way: disagreement implies different programs
- **Cannot obtain exact quantum state**
  - Measurement only yields a basis state
  - Disagreement does **not** imply different programs
  - Approximation by executing many times
    - **Expensive and no guarantee**
- Only checked for **some** quantum state
  - Uncountably many quantum states to test with

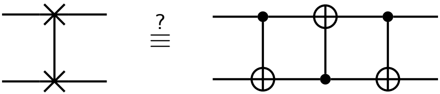
White-box equivalence check: Matrix representation



$$\stackrel{?}{\equiv}$$



White-box equivalence check: Matrix representation



$$\begin{bmatrix}1&0&0&0\\0&0&1&0\\0&1&0&0\\0&0&0&1\end{bmatrix}=\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}\begin{bmatrix}1&0&0&0\\0&0&0&1\\0&0&1&0\\0&1&0&0\end{bmatrix}\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}$$

- Matrices are **exponentially large** (n qubits $\rightsquigarrow 2^n \times 2^n$)

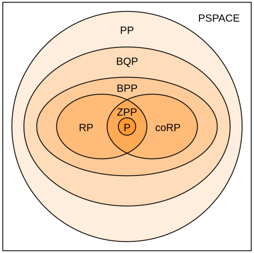
Computability

- Quantum computers and classical computers can solve **the same problems**
  - May be surprising because quantum gates are reversible
  - Simulation on quantum computer requires “garbage qubits”
  - Simulation on classical computer just multiplies matrices
- Advantage only comes in terms of **complexity**

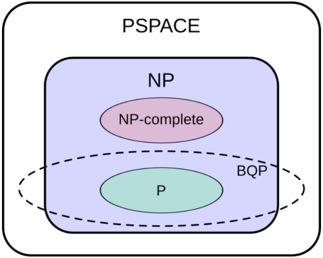
Complexity

- **P** = deterministic polynomial time
- **BPP** = bounded-error probabilistic polynomial time (error $< 1/3$)
- **BQP** = bounded-error quantum polynomial time (error $< 1/3$)
- **PP** = probabilistic polynomial time (error $< 1/2$)
- **NP** = nondeterministic polynomial time
- **PSPACE** = polynomial space

known:

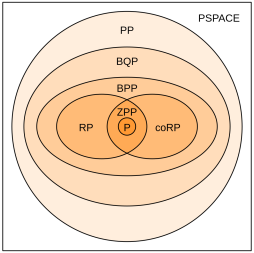


conjectured:



Complexity of simulating a quantum program

- Non-entangled qubits: only 2×2 matrices
- Clifford gates (Hadamard, CNOT, phase gate S) can be simulated in polynomial time[1]



- Generally, error-bounded probabilistic simulation of n qubits and m gates is possible with $\mathcal{O}(2^n m^3)$ classical gates[2]
- Simulation is BQP-complete
- Corollary: Simulation only requires polynomial space (follows from BQP $\subseteq$ PSPACE)

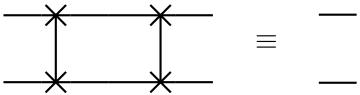
[1] D. Gottesman. PhD thesis. 1997

[2] R. Cleve. *Quantum Computation and Quantum Information Theory*. 2000.

Overview

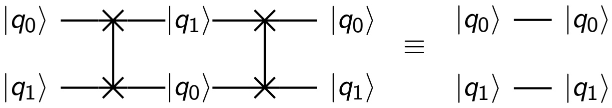
Toward a concise representation of quantum programs

- Intuitive explanation for the following equivalence?



Toward a concise representation of quantum programs

- Intuitive explanation for the following equivalence?

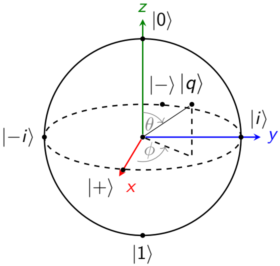


$$
\begin{aligned}
& SWAP(SWAP(|q_0, q_1\rangle)) \\
= \ & SWAP(|q_1, q_0\rangle) \\
= \ & |q_0, q_1\rangle
\end{aligned}
$$

Symbolic encoding of quantum programs

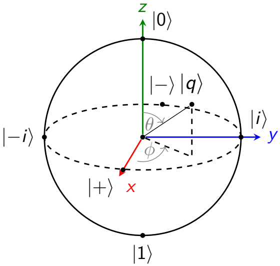
- Idea: use a symbolic (= logic) encoding
- Sketched for the main components

Qubit encoding



- $|q\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$
- $|\alpha|^2 + |\beta|^2 = 1$?

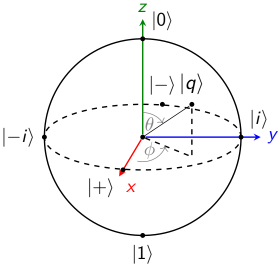
Qubit encoding



- $|q\rangle = \alpha |0\rangle + \beta |1\rangle$
- $|\alpha|^2 + |\beta|^2 = 1 \leftarrow$ **expensive**
- Instead use polar coordinates:

$$|q\rangle = \underbrace{\cos\frac{\theta}{2}}_{\alpha} |0\rangle + \underbrace{e^{i\phi} \sin\frac{\theta}{2}}_{\beta} |1\rangle$$

Qubit encoding



- $|q\rangle = \alpha |0\rangle + \beta |1\rangle$
- $|\alpha|^2 + |\beta|^2 = 1 \;\leftarrow$ **expensive**
- Instead use polar coordinates:

$$|q\rangle = \underbrace{\cos\frac{\theta}{2}}_{\alpha} |0\rangle + \underbrace{e^{i\phi}\sin\frac{\theta}{2}}_{\beta} |1\rangle$$

- Encode a qubit $|q\rangle$ as a 5-tuple $(\alpha, \beta_R, \beta_I, \phi, \theta) \in \mathbb{R}^5$
- Add constraints for values
  - $\alpha = \cos\frac{\theta}{2} \quad \wedge \quad \beta_R = \cos\phi \cdot \sin\frac{\theta}{2} \quad \wedge \quad \beta_I = \sin\phi \cdot \sin\frac{\theta}{2}$
  - $0 \leq \theta \leq \pi \quad \wedge \quad 0 \leq \phi < 2\pi$
  - $\theta = 0 \implies \phi = 0 \quad \wedge \quad \theta = \pi \implies \phi = 0$

Gate encoding

- Common gates can be encoded efficiently in a symbolic way
  - $I(|q_0\rangle) = |q_0\rangle$
  - $X(\alpha|0\rangle + \beta|1\rangle) = \beta|0\rangle + \alpha|1\rangle$
  - $H(\alpha|0\rangle + \beta|1\rangle) = \frac{1}{\sqrt{2}}(\alpha + \beta)|0\rangle + \frac{1}{\sqrt{2}}(\alpha - \beta)|1\rangle)$
  - $SWAP(|q_0, q_1\rangle) = |q_1, q_0\rangle$

Gate encoding

- Common gates can be encoded efficiently in a symbolic way
  - $I(|q_0\rangle) = |q_0\rangle$
  - $X(\alpha|0\rangle + \beta|1\rangle) = \beta|0\rangle + \alpha|1\rangle$
  - $H(\alpha|0\rangle + \beta|1\rangle) = \frac{1}{\sqrt{2}}(\alpha+\beta)|0\rangle + \frac{1}{\sqrt{2}}(\alpha-\beta)|1\rangle)$
  - $SWAP(|q_0, q_1\rangle) = |q_1, q_0\rangle$

  - $CNOT(|q_0, q_1\rangle)$?

Gate encoding

- Common gates can be encoded efficiently in a symbolic way
  - $I(|q_0\rangle) = |q_0\rangle$
  - $X(\alpha\,|0\rangle + \beta\,|1\rangle) = \beta\,|0\rangle + \alpha\,|1\rangle$
  - $H(\alpha\,|0\rangle + \beta\,|1\rangle) = \frac{1}{\sqrt{2}}(\alpha + \beta)\,|0\rangle + \frac{1}{\sqrt{2}}(\alpha - \beta)\,|1\rangle)$
  - $SWAP(|q_0, q_1\rangle) = |q_1, q_0\rangle$
  - $CNOT(|q_0, q_1\rangle)$?
- In general, we need an exponential representation

Measurement encoding

- Typically last operation
- Can be skipped with symbolic analysis
- Just a projection

Soundness and completeness

Theorem

The quantum program model (= our encoding) preserves the semantics of the quantum circuit model (= standard model)

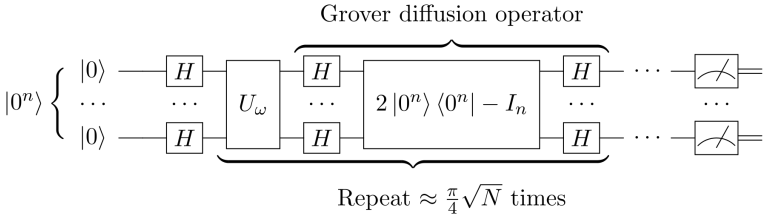
Corollary

Given a quantum program model with encoding $\mathcal{E}$ and a specification φ, the program satisfies φ if and only if $\mathcal{E} \wedge \neg\varphi$ is **unsatisfiable**

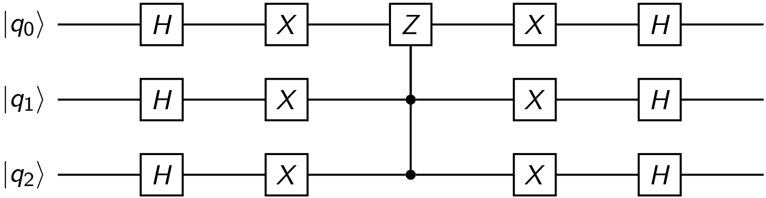
- $\mathcal{E}$ is a formula containing nonlinear real arithmetic with trigonometric expressions
- Undecidable but δ-decidable[1]
  - Implies that answer "**unsatisfiable**" is **correct**

[1] S. Gao, J. Avigad, and E. M. Clarke. *IJCAR*. 2012.

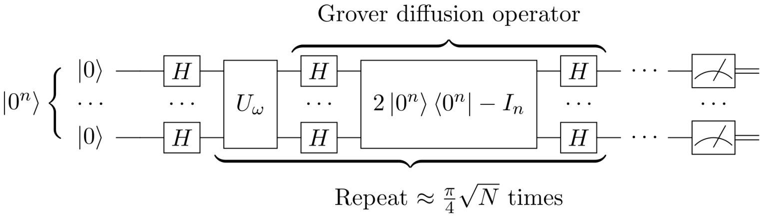
Example: Grover's diffusion operator



Example for $n = 3$:



Example: Grover's diffusion operator

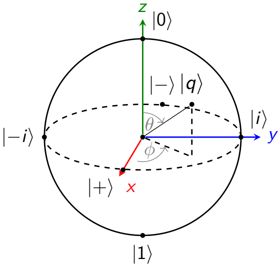


Specification:

- Each qubit with non-positive phase (α_i) reduces the phase

```
conjunction = []
for i in range(n):
   conjunction.append(Implies(
      initial_state[i].r <= 0,
      final_state[i].r <= initial_state[i].r))
```

Recall: Qubit encoding



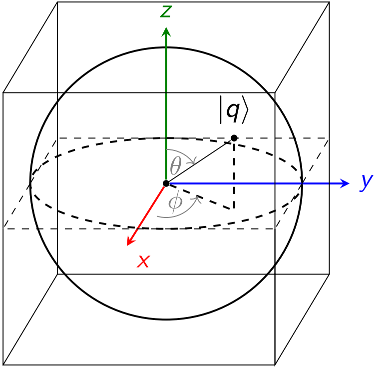
- $|q\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$
- $|\alpha|^2 + |\beta|^2 = 1 \;\leftarrow$ **expensive**
- Instead use polar coordinates:

$$|q\rangle = \underbrace{\cos\frac{\theta}{2}}_{\alpha}|0\rangle + \underbrace{e^{i\phi}\sin\frac{\theta}{2}}_{\beta}|1\rangle$$

- Encode a qubit $|q\rangle$ as a 5-tuple $(\alpha, \beta_R, \beta_I, \phi, \theta) \in \mathbb{R}^5$
- Add constraints for values
  - $\alpha = \cos\frac{\theta}{2} \quad\wedge\quad \beta_R = \cos\phi\cdot\sin\frac{\theta}{2} \quad\wedge\quad \beta_I = \sin\phi\cdot\sin\frac{\theta}{2}$
  - $0 \leq \theta \leq \pi \quad\wedge\quad 0 \leq \phi < 2\pi$
  - $\theta = 0 \implies \phi = 0 \quad\wedge\quad \theta = \pi \implies \phi = 0$

Overapproximation

- Exact initial-state constraints generally not needed
- **Positive** verification result for **supersets** sufficient



$$-1 \leq \alpha \leq 1 \quad \wedge \quad -1 \leq \beta_R \leq 1 \quad \wedge \quad -1 \leq \beta_I \leq 1$$

Encoding of motivating problems

- Correctness analysis $\mathcal{E} \models \varphi$ (seen before)
- Program equivalence
- Repair of errors and program synthesis
- Optimization (e.g., number of gates, types of gates, physical implementation adhering to constraints)

Encoding of motivating problems

- Correctness analysis $\mathcal{E} \models \varphi$ (seen before)
- Program equivalence

$$\forall x : \mathcal{E}_1(x) = \mathcal{E}_2(x)$$

- Repair of errors and program synthesis
- Optimization (e.g., number of gates, types of gates, physical implementation adhering to constraints)

Encoding of motivating problems

- Correctness analysis $\mathcal{E} \models \varphi$ (seen before)
- Program equivalence

$$\forall x : \mathcal{E}_1(x) = \mathcal{E}_2(x)$$

- Repair of errors and program synthesis

$$\exists G_1, \ldots, G_n : \text{circuit}(G_1 \ldots G_n) \models \varphi$$

- Optimization (e.g., number of gates, types of gates, physical implementation adhering to constraints)

Encoding of motivating problems

- Correctness analysis $\mathcal{E} \models \varphi$ (seen before)
- Program equivalence

$$\forall x : \mathcal{E}_1(x) = \mathcal{E}_2(x)$$

- Repair of errors and program synthesis

$$\exists G_1, \ldots, G_n : \text{circuit}(G_1 \ldots G_n) \models \varphi$$

- Optimization (e.g., number of gates, types of gates, physical implementation adhering to constraints)

$$\exists G_1, \ldots, G_n : \text{circuit}(G_1 \ldots G_n) \equiv \mathcal{E}$$

Overview

Benchmark problems

Program	Description	Depth	Input
Toffoli	Toffoli gate	5	bit vector
TP	Quantum teleportation	6	infinite
ADD-8	8-qubit quantum adder	48	bit vector
QFT-n	n-qubit quantum Fourier transform	$\mathcal{O}(n^2)$	bit vector
QPE-n	n-qubit quantum phase estimation	$\mathcal{O}(n^2)$	singleton[1]
GDO-n	n-qubit Grover's diffusion operator	$\mathcal{O}(n)$	infinite

[1] Parameterized gates

Algorithms

- **Simulation** on a classical computer
- **Matrix**: Encoding with (exponential) matrix/vector representation
- **Mapping**: Encoding with gate mapping but without overapproximation
- **Approx**: Encoding with gate mapping and overapproximation

Benchmark results

Benchmark	Simulation	Matrix	Mapping	Approx
Toffoli	**0.02 sec**	11.1 sec	1.3 sec	0.4 sec
TP	N/A	44.8 sec	**21.6 sec**	31.0 sec
ADD-8	6.1 h	`OOM`	**7.6 sec**	7.8 sec
QFT-3	**0.005 sec**	12.8 sec	5.8 sec	1.0 sec
QFT-5	**0.03 sec**	17.6 min	2.6 min	26.4 sec
QFT-10	**1.5 sec**	1.2 h	10.9 h	1.6 h
QFT-12	**14.0 sec**	4.0 h	`timeout`	7.4 h

Benchmark results

Benchmark	Simulation	Matrix	Mapping	Approx
QPE-3	N/A	19.2 sec	34.0 sec	**8.7 sec**
QPE-5	N/A	18.2 min	42.3 min	**3.9 min**
GDO-5	N/A	`timeout`	9.2 sec	**1.3 sec**
GDO-10	N/A	`timeout`	3.2 min	**17.0 sec**
GDO-12	N/A	`timeout`	14.2 min	**20.2 sec**
GDO-15	N/A	`timeout`	2.9 h	**1.0 min**
GDO-18	N/A	`timeout`	`timeout`	**4.9 min**
GDO-20	N/A	`timeout`	`timeout`	**17.1 min**
GDO-22	N/A	`timeout`	`timeout`	**1.1 h**
GDO-24	N/A	`timeout`	`timeout`	**4.2 h**

Overview

Conclusion and future work

- Symbolic encoding of quantum programs
- Fully automatic verification via δ-satisfiability
- Symbolic encoding can sometimes avoid exponential blow-up
- Simple overapproximation sometimes useful in practice
- Future directions:
  - Other approximation techniques
  - Falsification and approximation refinement

Demo

- Tool available at `https://github.com/schillic/symQV`
- Toffoli gate / CCNOT: $|q_0, q_1, q_2\rangle \mapsto |q_0, q_1, q_0 q_1 \oplus q_2\rangle$
- Universal gate for classical circuits

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Demo

- Tool available at https://github.com/schillic/symQV
- Toffoli gate / CCNOT: $|q_0, q_1, q_2\rangle \mapsto |q_0, q_1, q_0 q_1 \oplus q_2\rangle$
- Universal gate for classical circuits