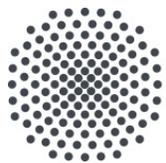


Quantum Applications in the NISQ Era



University of Stuttgart

Marie Salm

salm@iaas.uni-stuttgart.de

Institute of Architecture of Application Systems



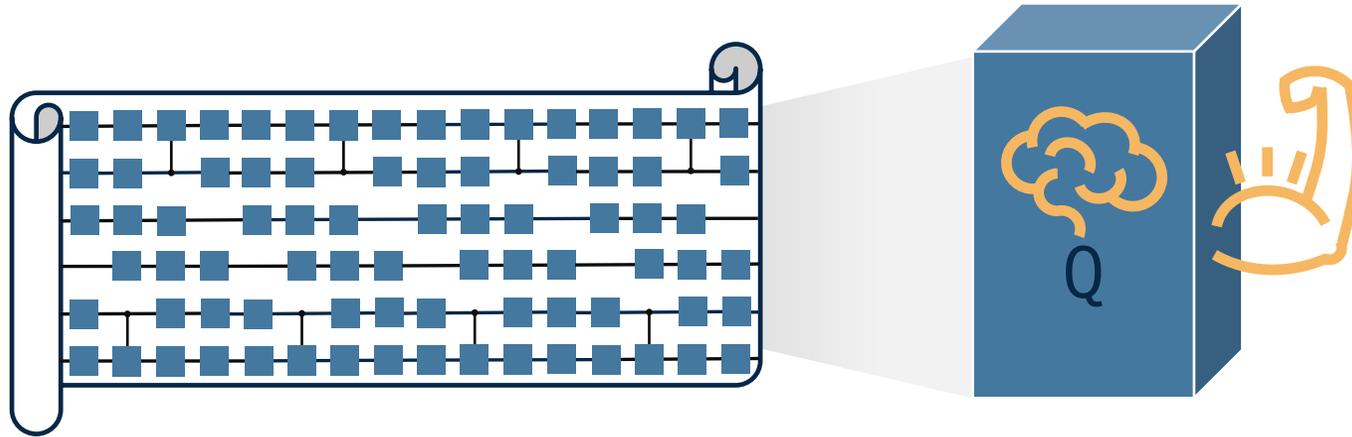
Supported by:



Federal Ministry
for Economic Affairs
and Climate Action

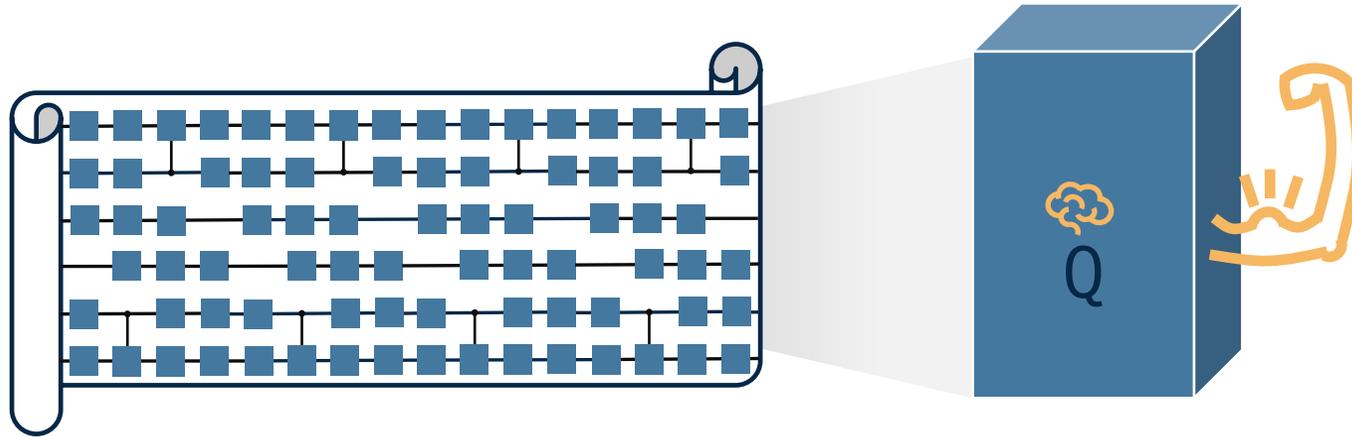
on the basis of a decision
by the German Bundestag

Future vs. today



- QPUs will solve problems faster

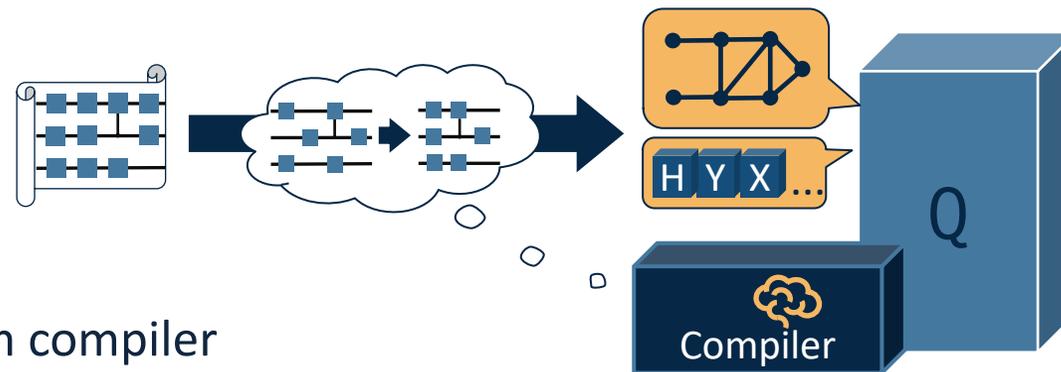
Future vs. today



- QPUs will solve problems faster
- **currently:** prone to errors & small qubit counts
 - determine **executable circuit sizes**, i.e., width & depth
 - design quantum circuits as **resource efficient** as possible

Influencing circuit size

- **true size** of circuit depends on
 - qubit topology
 - implemented gate set
 - mapping & optimization by quantum compiler

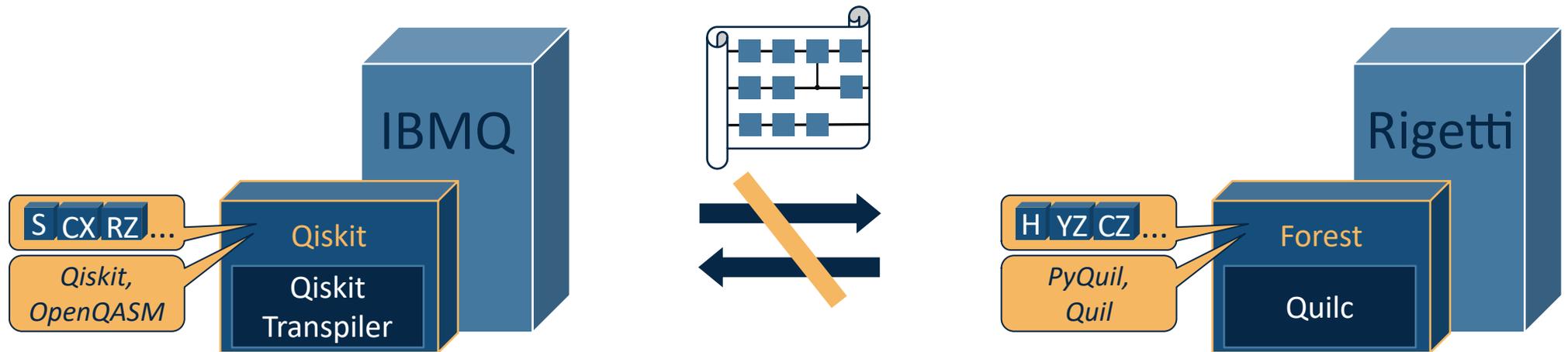


- **variety** of quantum compilers exists
- given a circuit: which compiler delivers **“best”** compilation result?
- comparison is required ...



Usage of quantum compilers

- most compilers **accessible via SDKs** differing in
 - gate sets
 - programming languages
 - vendors of QPUs



Analysis of quantum compilers and their SDKs [1]

- not every SDK supports OpenQASM

- many vendor-specific SDKs only support **own QPUs**

- many SDKs support custom QPUs but **not custom target gate sets**

- formats for defining custom QPUs differ

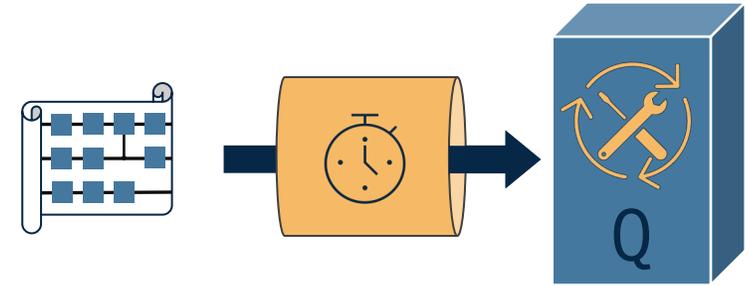
- SDK has to support the target QPU natively

	SDKs (Compilers)	Import Languages	Export Languages	Backend Vendors	Custom QPUs	Custom Compilation Gate Set
Propr. SDKs	Cirq	OpenQASM, Cirq-JSON, Quirk-JSON	OpenQASM, Quil, Cirq-JSON	Google, AQT, Pasqal	yes	yes
	Forest (Quilc)	PyQuil*, Quil	Quil	Rigetti	yes	no
	Qiskit (Transpiler)	OpenQASM, Qiskit*	OpenQASM	IBM	yes	yes
Independent SDKs	pytket (tket)	OpenQASM, Qiskit*, PyZX*, PyQuil*, Cirq*, Quipper	OpenQASM, Qiskit*, PyZX*, PyQuil*, Cirq*, Qulacs*, Q#, ProjectQ*	AQT, Amazon Braket, Honeywell, Rigetti, IBM, Microsoft QDK	yes	yes
	staq	OpenQASM	OpenQASM, ProjectQ*, Quil, Q#, Cirq*	no	no	no
	ProjectQ	ProjectQ*	ProjectQ*	IBM, AQT, Amazon Braket, IonQ	no	yes

*Python libraries

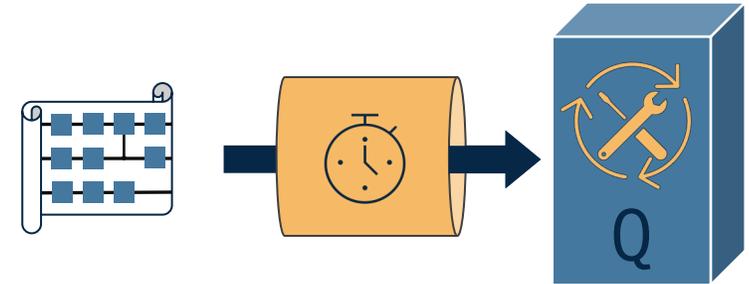
In addition ...

- QPUs are regularly re-calibrated
 - changes in error rates
- cloud providers regulate computing access
 - resulting in waiting times
- implementation & execution of circuit
 - QPU
 - compiler
 - SDK
 - access regulations



In addition ...

- QPUs are regularly re-calibrated
 - changes in error rates
- ... to computing access

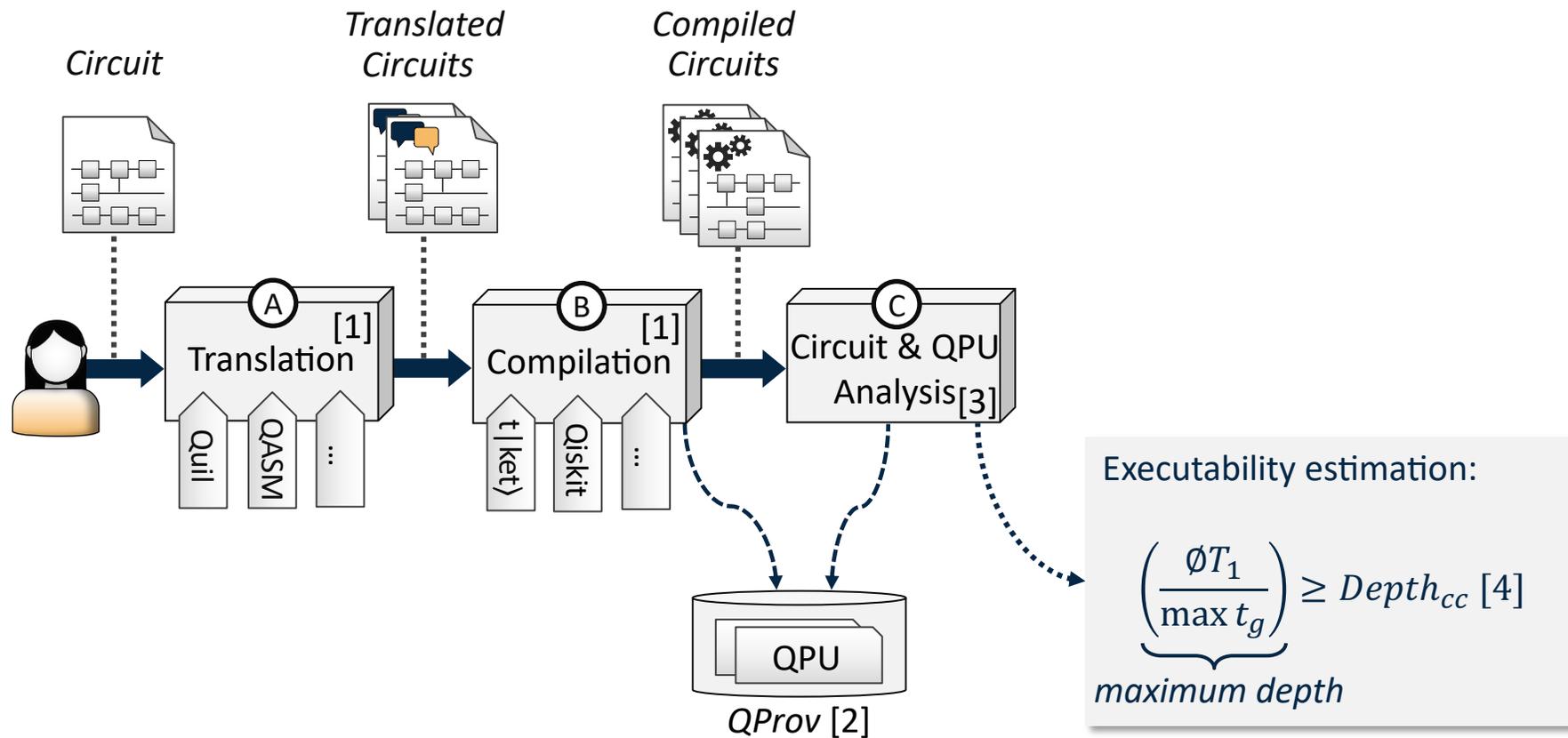


deep technical knowledge & huge manual effort required

- implementation & execution
 - QPU
 - compiler
 - SDK
 - access regulations

Approach

Approach – NISQ Analyzer



[1] Marie Salm, Johanna Barzen, Frank Leymann, Benjamin Weder, Karoline Wild. 2021. Automating the Comparison of Quantum Compilers for Quantum Circuits. In: Proceedings of the 15th Symposium and Summer School on Service-Oriented Computing (SummerSOC 2021), Springer International Publishing, 2021

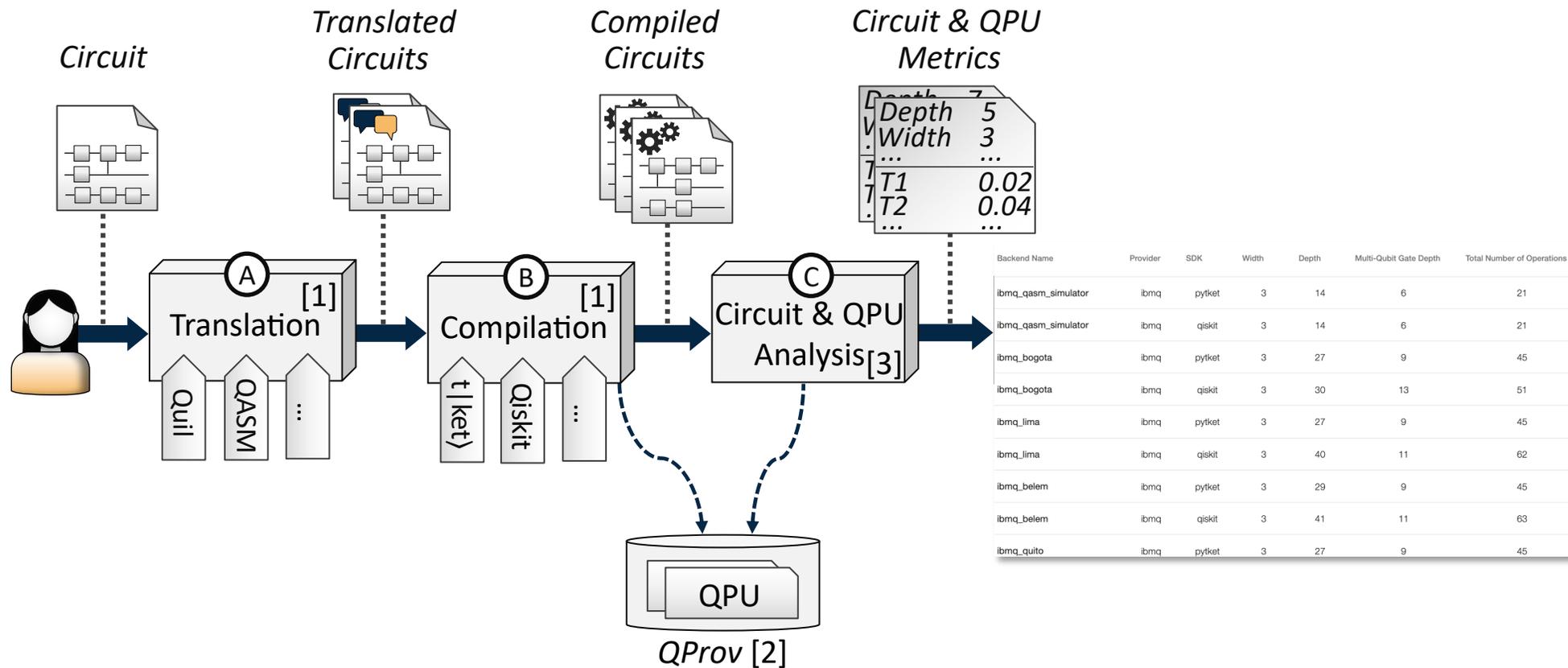
[2] Benjamin Weder, Johanna Barzen, Frank Leymann, Marie Salm, Karoline Wild. 2021. QProv: A provenance system for quantum computing. In: IET Quantum Communications, Wiley, 2021

[3] Marie Salm, Johanna Barzen, Uwe Breitenbücher, Frank Leymann, Benjamin Weder, et al. 2020. The NISQ Analyzer: Automating the Selection of Quantum Computers for Quantum Algorithms. In: Proceedings of the 14th Symposium and Summer School on Service-Oriented Computing (SummerSOC 2020), Springer International Publishing, 2020

[4] Eyob A. Sete, William J. Zeng, and Chad T. Rigetti. 2016. A functional architecture for scalable quantum computing. In 2016 IEEE International Conference on Rebooting Computing(ICRC).1–6.

[5] Marie Salm, Johanna Barzen, Frank Leymann, Benjamin Weder: Prioritization of Compiled Quantum Circuits for Different Quantum Computers. In: The 1st International Workshop on Quantum Software Analysis, Evolution and Reengineering (Q-SANER 2022) co-located with IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER), 2022. ©Marie Salm 10

Approach – NISQ Analyzer



[1] Marie Salm, Johanna Barzen, Frank Leymann, Benjamin Weder, Karoline Wild. 2021. Automating the Comparison of Quantum Compilers for Quantum Circuits. In: Proceedings of the 15th Symposium and Summer School on Service-Oriented Computing (SummerSOC 2021), Springer International Publishing, 2021

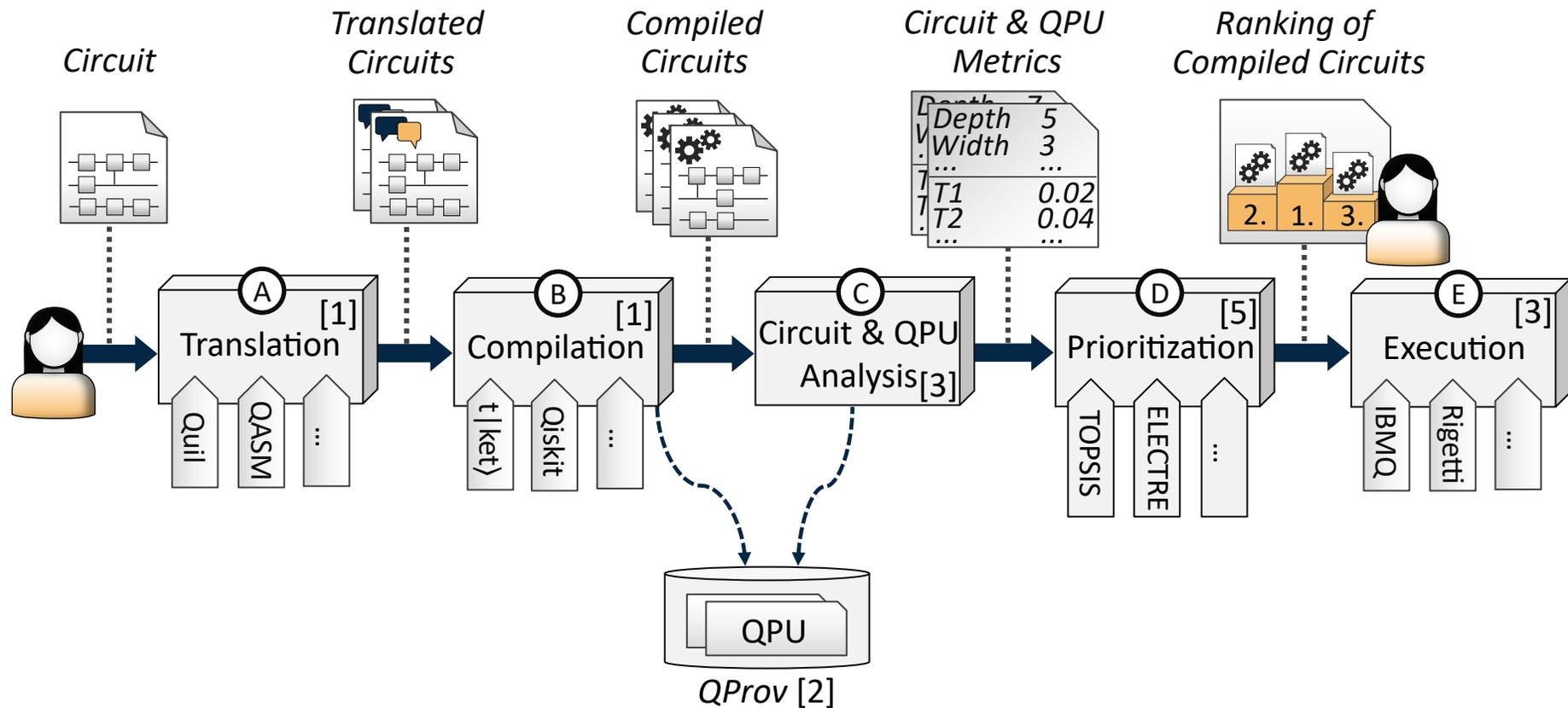
[2] Benjamin Weder, Johanna Barzen, Frank Leymann, Marie Salm, Karoline Wild. 2021. QProv: A provenance system for quantum computing. In: IET Quantum Communications, Wiley, 2021

[3] Marie Salm, Johanna Barzen, Uwe Breitenbücher, Frank Leymann, Benjamin Weder, et al. 2020. The NISQ Analyzer: Automating the Selection of Quantum Computers for Quantum Algorithms. In: Proceedings of the 14th Symposium and Summer School on Service-Oriented Computing (SummerSOC 2020), Springer International Publishing, 2020

[4] Eyob A. Sete, William J. Zeng, and Chad T. Rigetti. 2016. A functional architecture for scalable quantum computing. In 2016 IEEE International Conference on Rebooting Computing(ICRC).1–6.

[5] Marie Salm, Johanna Barzen, Frank Leymann, Benjamin Weder: Prioritization of Compiled Quantum Circuits for Different Quantum Computers. In: The 1st International Workshop on Quantum Software Analysis, Evolution and Reengineering (Q-SANER 2022) co-located with IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER), 2022. ©Marie Salm 11

Approach – NISQ Analyzer



[1] Marie Salm, Johanna Barzen, Frank Leymann, Benjamin Weder, Karoline Wild. 2021. Automating the Comparison of Quantum Compilers for Quantum Circuits. In: Proceedings of the 15th Symposium and Summer School on Service-Oriented Computing (SummerSOC 2021), Springer International Publishing, 2021

[2] Benjamin Weder, Johanna Barzen, Frank Leymann, Marie Salm, Karoline Wild. 2021. QProv: A provenance system for quantum computing. In: IET Quantum Communications, Wiley, 2021

[3] Marie Salm, Johanna Barzen, Uwe Breitenbücher, Frank Leymann, Benjamin Weder, et al. 2020. The NISQ Analyzer: Automating the Selection of Quantum Computers for Quantum Algorithms. In: Proceedings of the 14th Symposium and Summer School on Service-Oriented Computing (SummerSOC 2020), Springer International Publishing, 2020

[4] Eyob A. Sete, William J. Zeng, and Chad T. Rigetti. 2016. A functional architecture for scalable quantum computing. In 2016 IEEE International Conference on Rebooting Computing(ICRC).1–6.

[5] Marie Salm, Johanna Barzen, Frank Leymann, Benjamin Weder: Prioritization of Compiled Quantum Circuits for Different Quantum Computers. In: The 1st International Workshop on Quantum Software Analysis, Evolution and Reengineering (Q-SANER 2022) co-located with IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER), 2022. ©Marie Salm 12

Prototype

List of algorithms



Search



<input type="checkbox"/>	Name	Acronym	Type	Problem
<input type="checkbox"/>	Anomaly Detection with QGANs		HYBRID	The goal for this problem is an automate...
<input type="checkbox"/>	Dürr-Høyer Quantum Minimization Algorit...		HYBRID	Given a function $f(x): \{0, 1\}^n \rightarrow \mathbb{R}$
<input type="checkbox"/>	Grover 	Grover Algorithm	QUANTUM	
<input type="checkbox"/>	HHL Algorithm		QUANTUM	A classical linear system of equations is ...
<input type="checkbox"/>	Inverse quantum Fourier transform		QUANTUM	Given the following quantum state on an ...
<input type="checkbox"/>	Quantum Annealing	QA	QUANTUM	Since QA is a generic metaheuristic, it ca...
<input type="checkbox"/>	Quantum Annealing of the Knapsack Pro...		QUANTUM	In the NP-Complete Knapsack Problem, ...
<input type="checkbox"/>	Quantum approximate optimization algor...	QAOA	HYBRID	QAOA solves binary optimization proble...

Implementation of an algorithm



Grover (Grover Algorithm)

Quantum Algorithm

General

Implementations

Related algorithms

Publications



<input type="checkbox"/>	Name	Description	Dependencies
<input type="checkbox"/>	Grover SAT	The implementation is provided by Qiskit The impleme...	

Items per page:

10



NISQ Analyzer



Grover / Grover SAT

Quantum Algorithm Quantum Implementation



General

Publications

Software Platforms

Selection Criteria

Execution

NISQ Analyzer

QPU Analysis Jobs

New Analysis

Time

Readiness

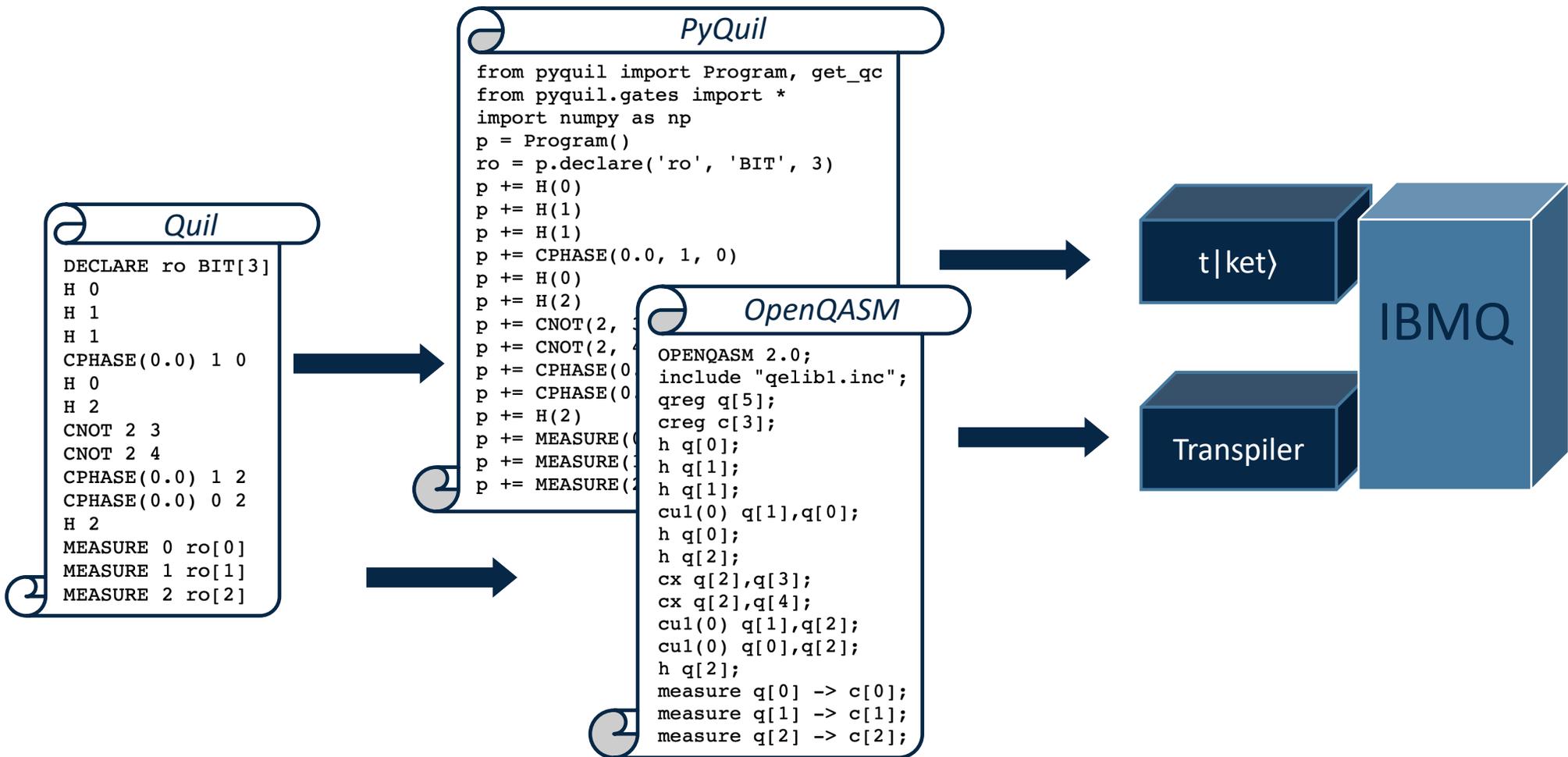
2022-05-18T20:50:40.44Z

Show analysis

Start analysis

The screenshot shows the QC Atlas web interface. At the top, there is a dark blue header with the QC Atlas logo and name. Below the header, the main content area is titled 'Grover / Grover SAT' with sub-headers 'Quantum Algorithm' and 'Quantum Implementation'. A navigation bar contains tabs for 'General', 'Publications', 'Software Platforms', 'Selection Criteria', 'Execution', and 'NISQ Analyzer'. The 'NISQ Analyzer' tab is active. On the left, there is a section titled 'QPU Analysis Jobs' with a 'New Analysis' button. Below this, there is a table with columns for 'Time' and 'Readiness'. A modal dialog box titled 'Start QPU-Selection-Analysis' is open in the center. The dialog has a 'Vendor' dropdown menu set to 'IBMQ', a 'Token *' input field, and a section for 'Select SDKs to be used for compilation:' with checkboxes for 'pytket', 'qiskit', and 'Include Simulators'. At the bottom of the dialog are 'Cancel' and 'Ok' buttons.

Implementation in Quil for IBMQ



Compilation results



Grover / Grover SAT

Quantum Algorithm / Quantum Implementation

General Publications Software Platforms Selection Criteria Execution **NISQ Analyzer**

Back

Prioritize

QPU Analysis Job from 2022-05-18T20:50:40.44Z

Check if QPU data is up-to-date ↻

Rank	Score	Backend Name	Provider	SDK	Width	Depth	Multi-Qubit Gate Depth	Total Number of Operations	Number of Single-Qubit Gates
-	-	ibmq_qasm_simulator	ibmq	pytket	5	20	6	40	32
-	-	ibmq_qasm_simulator	ibmq	qiskit	5	56	29	79	46
-	-	ibmq_santiago	ibmq	pytket	5	135	67	162	91
-	-	ibmq_santiago	ibmq	qiskit	5	114	69	171	95
-	-	ibmq_bogota	ibmq	pytket	5	127	57	170	103
-	-	ibmq_bogota	ibmq	qiskit	5	113	69	171	95
-	-	ibmq_lima	ibmq	pytket	5	124	63	161	91

Prioritization of compiled circuits



Grover / Grover SAT

Quantum Algorithm / Quantum Implementation

General Publications Software Platforms Selection Criteria Execution **NISQ Analyzer**

Back

Prioritize

QPU Analysis Job from 2020-05-10T00:50:40-117

Check if QPU data is up-to-date

Rank	Score	Backend Name	Multi-Qubit Gate Depth	Total Number of Operations	Number of Single-Qubit Gates
-	-	ibmq_qasm_simulator	6	40	32
-	-	ibmq_qasm_simulator	29	79	46
-	-	ibmq_santiago OUTDATED	67	162	91
-	-	ibmq_santiago OUTDATED	69	171	95
-	-	ibmq_bogota OUTDATED	57	170	103
-	-	ibmq_bogota OUTDATED	69	171	95

Prioritize Analysis Results

Select predefined preferences

- Short waiting time
- Stable execution results

Advanced settings

Or define individual metric weights

Cancel Ok

Ranked list of compilation results



General

Publications

Software Platforms

Selection Criteria

Execution

NISQ Analyzer

Back

Prioritize

Analyze Rank Sensitivity

QPU Analysis Job from 2022-05-18T20:50:40.44Z

Rank	Score	Backend Name	Provider	SDK	Number of Measurement Operations	Avg. Single-Qubit Gate Error	Avg. Multi-Qubit Gate Error	Avg. Single-Qubit G
1	1	ibmq_qasm_simulator	ibmq	pytket	2	0	0	0
2	1	simulator_statevector	ibmq	pytket	2	0	0	0
3	0.7443880	ibmq_qasm_simulator	ibmq	qiskit	2	0	0	0
4	0.7443880	simulator_mps	ibmq	qiskit	2	0	0	0
5	0.7443880	simulator_statevector	ibmq	qiskit	2	0	0	0
6	0.6816328	simulator_extended_stabilizer	ibmq	qiskit	2	0	0	0
7	0.2443449	ibmq_quito OUTDATED	ibmq	pytket	2	0.00032977032	0.25796703	26.6666
8	0.2409065	ibmq_quito OUTDATED	ibmq	qiskit	2	0.00032977032	0.25796703	26.6666
9	0.2071411	ibmq_lima	ibmq	pytket	2	0.00025296074	0.011037594	26.6666
10	0.2060959	ibmq_manila OUTDATED	ibmq	pytket	2	0.00021612592	0.006822977	26.6666

Execution



General

Publications

Software Platforms

Selection Criteria

Execution

NISQ Analyzer

Back

Prioritize

QPU Analysis Job from 2022-05-18T20:50:40.44Z

Rank	Score	Backend Name	Provider	SDK	-Qubit Gate Time (ns)	Avg. Readout Error	Avg. T1 (ns)	Avg. T2 (ns)	Queue Size	Execution Result
-	-	ibmq_qasm_simulator	ibmq	pytket	0	0	-	-	7	Show result
-	-	ibmq_qasm_simulator	ibmq	qiskit	0	0	-	-	7	Execute
-	-	ibmq_santiago OUTDATED	ibmq	pytket	391.1111	0.03626	144426.28	108564.17	138	Execute
-	-	ibmq_santiago OUTDATED	ibmq	qiskit	391.1111	0.03626	144426.28	108564.17	138	Execute
-	-	ibmq_bogota OUTDATED	ibmq	pytket	487.1111	0.08396	125368.09	177174.05	101	Show result
-	-	ibmq_bogota OUTDATED	ibmq	qiskit	487.1111	0.08396	125368.09	177174.05	101	Execute
-	-	ibmq_lima	ibmq	pytket	387.55554	0.02722	91116.516	107485.53	141	Execute

Status: FINISHED

Number of shots: 1024

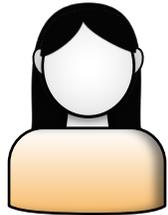
Histogram intersection value: 0.9375

Result: {00=224, 01=260, 10=265, 11=275}

Input-sensitive implementation selection

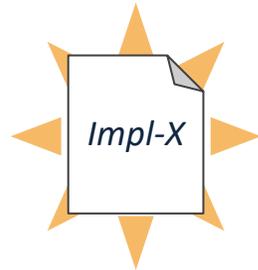
How to find implementations that can execute the input data?

Factorizing

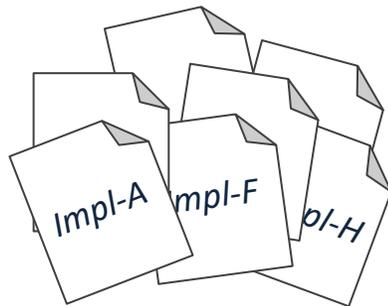


N = 255

Shor!



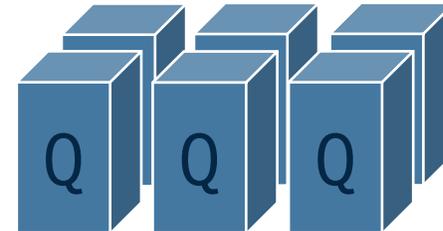
255 ok?



+

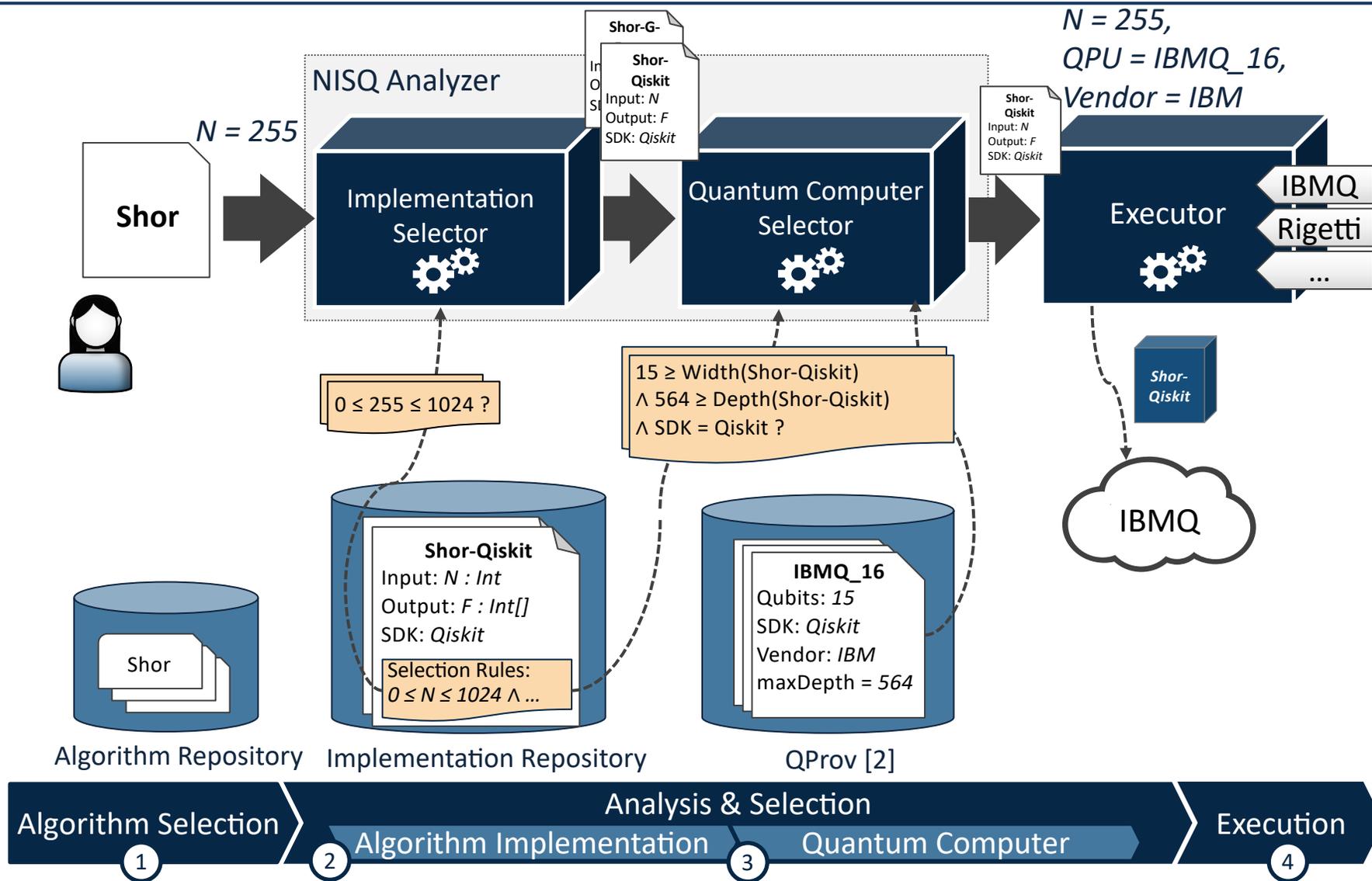


enough qubits, coherence time, ...?



immense effort & knowledge needed!

NISQ Analyzer – Implementation & QPU Selection [1]



[1] Marie Salm, Johanna Barzen, Uwe Breitenbücher, Frank Leymann, Benjamin Weder, et al. 2020. The NISQ Analyzer: Automating the Selection of Quantum Computers for Quantum Algorithms. In: Proceedings of the 14th Symposium and Summer School on Service-Oriented Computing (SummerSOC 2020), Springer International Publishing, 2020

[2] Weder, Benjamin; Barzen, Johanna; Leymann, Frank; Salm, Marie; Wild, Karoline: QProv: A Provenance System for Quantum Computing. In: IET Quantum Communications, Wiley, 2021

Analysis based on input data

The screenshot shows the QC Atlas interface for the Shor algorithm. The page has a dark blue header with the QC Atlas logo and the text 'Shor'. Below the header, there are tabs for 'General', 'Implementations', 'Related algorithms', 'Publications', 'NISQ Analyzer' (which is active), and 'NISQ Results'. A '+ Factorization' button is visible in the top right. The main content area is titled 'Analysis Jobs' and contains a 'New Analysis' button. A 'Start Analysis' dialog box is overlaid on the page, containing input fields for 'N', 'Cloud Service' (a dropdown menu), and 'Your Token', along with 'Cancel' and 'Ok' buttons. In the background, a table of analysis jobs is visible with columns for 'Time' and 'Readiness', and a 'Show analysis' button for each row.

	Time	Readiness
	2022-06-02T18:40:36.224119Z	Show analysis
	2022-06-02T19:05:57.263834Z	Show analysis
	2022-06-02T19:10:31.049747Z	Show analysis
	2022-06-02T19:45:33.816735Z	Show analysis
	2022-06-02T19:47:20.595699Z	Show analysis
	2022-06-02T19:54:22.10116Z	Show analysis

Compilation results of different implementations

Check if QPU data is up-to-date

Implementation: shor-fix

Backend Name	Provider	SDK	Width	Depth	Multi-Qubit Gate Depth	Total Number of Operations	Number of Single-Qubit Gates	Number of Multi-Qubit Gates
ibmq_qasm_simulator	ibmq	qiskit	5	7	4	10	2	5
ibmq_santiago	ibmq	qiskit	5	15	3	24	15	6
ibmq_bogota	ibmq	qiskit	5	17	8	26	12	11
ibmq_lima	ibmq	qiskit	4	12	5	14	6	5
ibmq_belem	ibmq	qiskit	5	12	7	17	6	8
ibmq_quito	ibmq	qiskit	5	12	7	17	6	8
simulator_statevector	ibmq	qiskit	5	7	4	10	2	5
simulator_mps	ibmq	qiskit	5	6	4	9	1	5
simulator_extended_stabilizer	ibmq	qiskit	3	7	2	9	4	2
ibmq_manila	ibmq	qiskit	5	11	5	16	5	8

Implementation: shor-general

Backend Name	Provider	SDK	Width	Depth	Multi-Qubit Gate Depth	Total Number of Operations	Number of Single-Qubit Gates	Number of Multi-Qubit Gates
ibmq_qasm_simulator	ibmq	qiskit	5	7	4	10	2	5
ibmq_santiago	ibmq	qiskit	4	18	6	25	16	6
ibmq_lima	ibmq	qiskit	4	12	5	14	6	5

Conclusion

Conclusion

- to select/design & execute a circuit
 - QPU, compiler, SDK & access must be considered
 - for precise & fast executions
- automated selection, translation, compilation, prioritization & execution
 - plug-in based, supports extensibility
- in the future
 - further SDKs
 - deeper analyzes
 - integration into QHAna

Thank you!