DFN Messages

Finely orchestrated

DFN Security Operations in the Overall Composition

Support online teaching

new DFNconf

Agreements

So sovereign

digital identities &

Trust Services

9 770177 689001

Imprint

Publisher: Association for the Promotion of a
German Research Network

DFN Association
Alexanderplatz 1, 10178 Berlin
Phone: 030 - 88 42 99 - 0
Fax: 030 - 88 42 99 - 370
Mail: presse@dfn.de Web:
www.dfn.de

ISSN 0177-6894

Editors: Maimona Id, Nina Bark
Proofreading: Angela Lenz
Design: Labor3 | www.labor3.com
Printing: Druckerei Rüss, Potsdam ©
DFN-Verein 06/2022

Photo credits
Title: trodler / freepik
Back: bruno135_406, nojustice / Adobe Stock

Prof. Dr. Matthias S. Müller
Chair of High
Performance Computing and
Director of the IT Center of the
Rhenish-Westphalian
RWTH Aachen University of Applied Sciences

The open exchange and the joint use of resources are high values in the scientific community. This is currently reflected in the disciplinary and national
Open Science movement with the aim of sharing and reusing research data as well as in the worldwide development of distributed and jointly usable research infrastructures.

In contrast, we are facing ever greater challenges in comprehensively protecting this valuable capital in science and research. Increasingly complex cyber attacks are directly targeting the ability of universities and scientific institutions to act
facilities and cause devastating damage. Only through increased efforts can we increase protection and reduce damage. In addition to mere prevention, the focus has therefore shifted to rapid detection and adequate response to security incidents.

"Safety is not a state, but a process," as the saying goes. Part of this process, which is anchored in information security management, is to continuously review, question, improve and, if necessary, supplement one's own security measures. An important criterion for the appropriateness of measures

is the state of the art, located between the proven recognized rules of technology and the further developed state of the art. The state of science of yesterday is the state of the art of today – and what is still state of the art today will tomorrow only be a recognized rule of technology. The members of the DFN Association are active shapers of this development.

Information security is a task to which we must commit ourselves across all institutions. This requires a common understanding of information security processes and the technologies and measures required for them. It is important to pool resources, provide skills and share information. The scientific institutions organized in the DFN-Verein are taking on this task, for example, by testing the DFN-Verein's new "Security Operations" in pilot operation together with the office and DFN-CERT and preparing them for regular operation.

In the future, security operations will become an important component of the information security of all participants in the DFN. If we succeed in efficiently implementing the state of the art across the board and if we also work together to continuously advance the state of the art, then we will achieve a lot in terms of information security!

Sincerely,
Matthias S. Müller

Quantum simulators in practice

If you want to take a closer look at quantum computing now, simulators will give you a good introduction to the world of qubits, quantum networks and tap-proof quantum protocols. Software-based simulators offer a wide range of possibilities not only for researchers, which can be used to simulate different areas of quantum technology.

Text: **Sascha Schweiger, Martin Seidel** (Regional Computing Center Erlangen, Friedrich-Alexander-University Erlangen-Nuremberg)

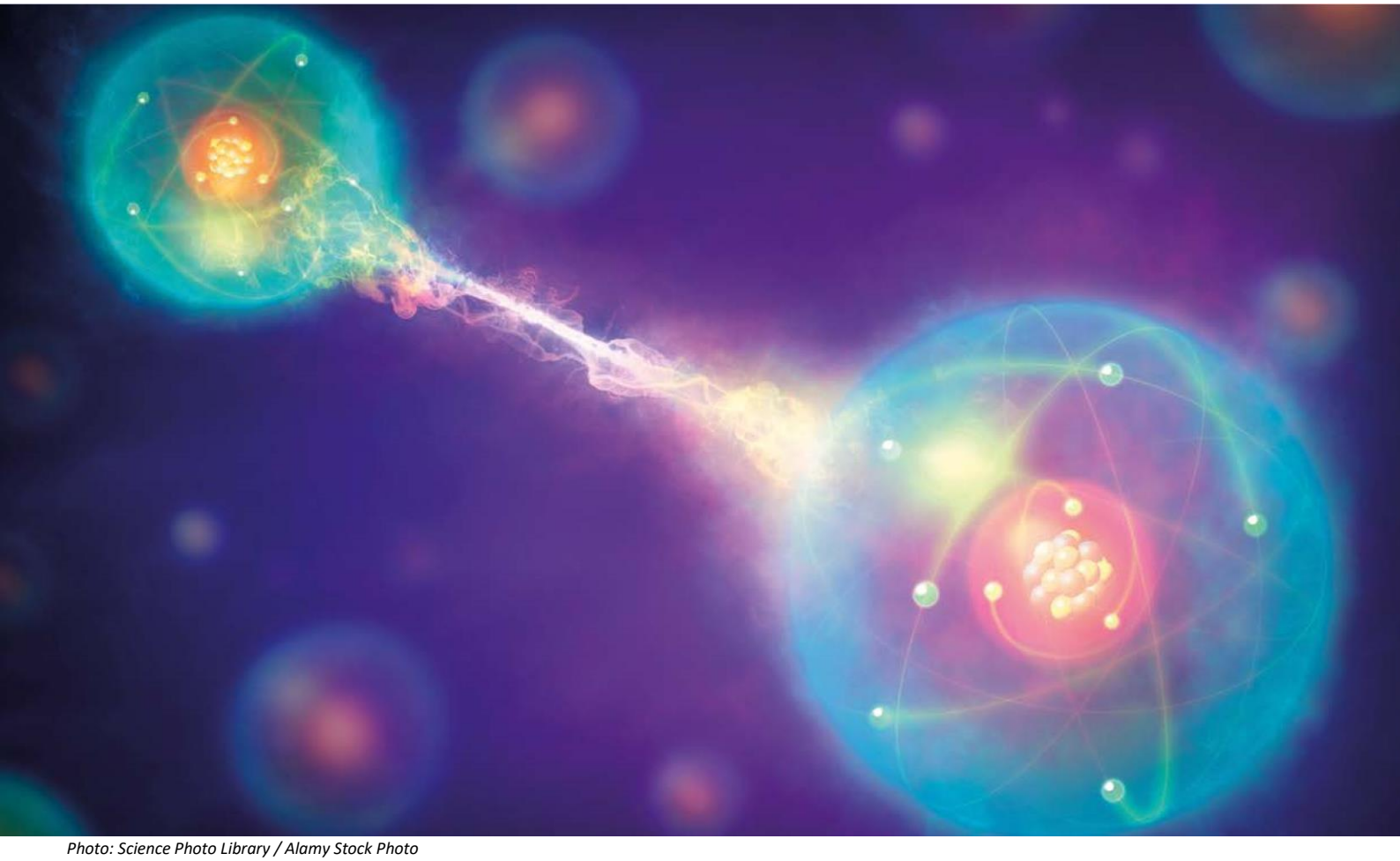
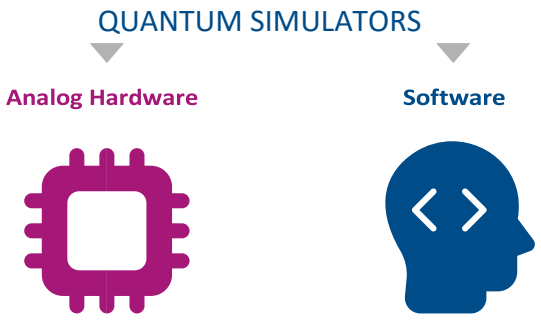


Photo: Science Photo Library / Alamy Stock Photo

The term and the field of quantum simulation are still relatively young, even if initial ideas seem to reach back until the understanding of Richard Feynman,
who recognized that the computing power of conventional computers is not sufficient to compute complex quantum systems, but that a "simpler" quantum system is needed as a "simulator" to replicate a much more complex system. An important step towards universal quantum computers.

Quantum simulators are often applied analogously with systems that attempt to recreate quantum effects through simpler, easier-to-control analog hardware systems. In contrast to these specially tailored hardware systems, this is about software-based simulators that can be used to replicate various areas of quantum technology. Such software can be used, for example, to simulate the physical plane with circuit elements and damping effects. However, there is also software that makes it possible to test routing, protocols and distributed key transmission in the network.



Software-based quantum simulators therefore already now offer the opportunity to develop and test algorithms and software solutions for future, universally programmable quantum computers. They can also be used to study the impact and influence of this technology on communication networks, cybersecurity and computing power.

> The computing power of conventional computers is not sufficient to calculate complex quantum systems.

In the field of quantum networks, they are used, for example, to simulate quantum repeaters. It`s still being worked upon Quantum repeaters. However, as soon as these are available as hardware, the findings from the simulations can accelerate the realization.

The different quantum simulators

Although there is no set of rules for categorization yet, quantum simulators can be divided into different classes. These include quantum circuit simulators, quantum network simulators and special application simulators (see table). Circuit simulators can simulate basic hardware components such as quantum memory, quantum gates, and quantum states such as entanglement and quantum teleportation. Network simulators often contain many of the functionalities of circuit simulators, but they are designed to study the influence and behavior of quantum technologies on networks and to develop protocols based on this technology. To the special

OVERVIEW OF THE ACCESS AND SPECIAL AREAS OF APPLICATION OF THE SIMULATORS

SIMULATOR	ACCESS	SIMULATOR ESPECIALLY SUITABLE FOR
IBM Quantum/Qiskit	web-based platform with quantum hardware/download	Circuits
Quantum Network Explorer	web-based platform	Networks
QKDSimulator	web-based platform	Special applications / QKD
SQUANCH	Download / Self-installation	Circuits, Networks
QuNetSim	Download / Self-installation	Networks
SeQUeNCe	Download / Self-installation	Special applications / hardware

Quantum simulators count simulators that have been developed for specific use cases, such as QKD (Quantum Key Distribution).

Simulations on Platforms

In the meantime, there are some highy comprehensive and freely accessible online platforms for quantum simulation that should simplify the introduction to the topic of quantum computing. These include, for example, IBM Quantum platform, Google AI and Braket from Amazon.

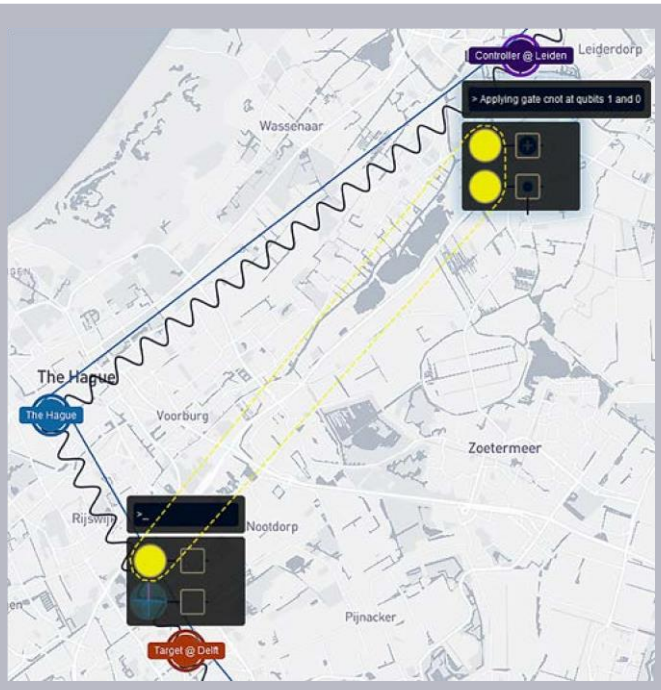


Figure 2: Surface of Quantum Network Explorer by QuTech

Already since 2019, the Leibniz Research (LRZ) offers a 42 qubit strong simulator – the SuperMUC NG – for quantum computing from Intel, albeit only for research purposes for now. The LRZ is also cooperating with Atos to make the "Atos Quantum Learning Machine" – the most commercially powerful simulator to date – and other helpful services available to researchers via the cloud.

Particularly suitable for distributed applications is the Quantum Network Explorer, an online tool for simulation and programming developed by QuTech. Since examples are already available, there is no need for installation and no need to program your own circuits. The tool is purely web-based and has a clear visualization (see Figure 2).

A network consisting of network nodes and network channels with adjustable gate and elementary fidelity can be used to transmit Qubits with complex states and thus simulate distributed applications. Intermediate steps are listed in detail and elaborately visualized. At the end, the results are clearly presented. Figure 2 shows the visualization of an intermediate step of a "state teleportation" between transmitter and receiver, which are connected to each other via an intermediate station. In this intermediate step, the entangled Qubits necessary for teleportation can be seen.

With the help of the IBM Quantum Simulator/Qiskit, it is possible to let plan, implement and emulate your first quantum circuits via a web browser. Those who have access can then let their simulations run already these days on real hardware with the IBM Quantum platform if they wish.

The IBM Quantum/Qiskit online platform can not only execute specially created quantum circuits on real quantum hardware, but also makes it possible to work exclusively in the web browser, as all the necessary tools are already available

> With the help of the IBM Quantum Simulator Qiskit, it is possible to plan the first quantum circuits via a web browser.

are on board. With the composer, circuits can be created without having to write codes yourself. The probability of the possible outcomes is clearly displayed in a histogram. These self-created circuits can be exported to a Jupyter notebook (Python) or used directly in the integrated WEB IDE. Jupyter Notebooks, formerly IPython Notebooks, are to be understood as interactive web documents or as an environment in their own open-source file format ".ipynb". The input and output are each done in separate cells, making it easier to manage text, plots and code in numerous languages. This helps, for example, in the evaluation of data. Figure 3 shows the Composer in the upper left corner, which can be used to create various circuits from classic

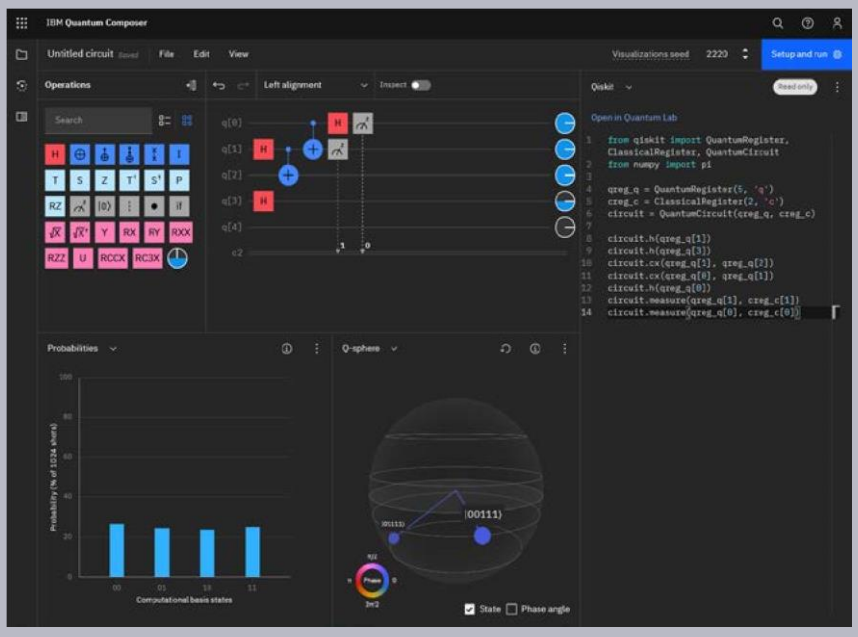


Figure 3: Web interface of the IBM Quantum/Qiskit simulation platform

Figure 4: User Interface for Configuration the parameter in the QKDSimulator

and quantum gates. At the same time as the circuit elements (left), the corresponding code is then generated (right). A histogram is generated in near real time (bottom left) and shows the distribution of all possible results of the simulation. The Q-Sphere (not to be confused with the Bloch sphere) is an alternative way of representing the histogram.

To test and visualize a quantum key distribution (BB84 protocol), the online simulator QKDSimulator can be used to configure parameters of the quantum channel for error matching and correction as well as error tolerance rates by sliders on a graphical user interface (Figure 4).

Simulators for self-installation

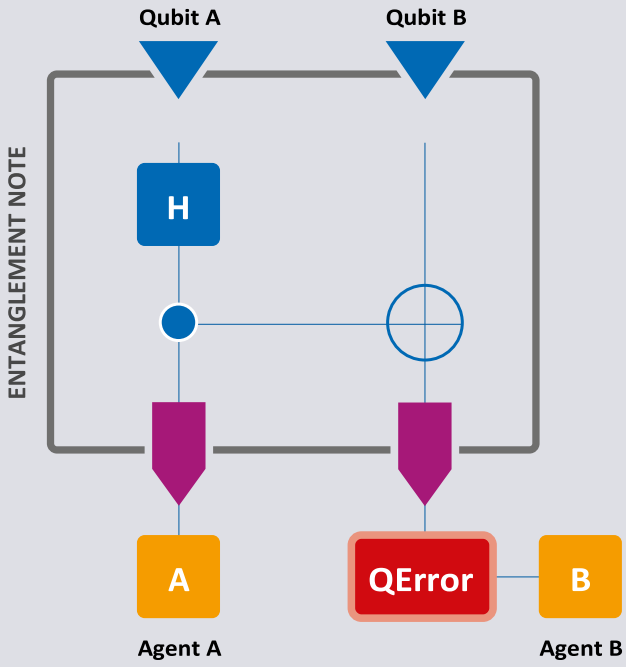
If you don't want to work with web-based platforms, but prefer to install your own simulation software, you can access various software packages. These packages are usually publicly available and suitable for different areas of application depending on the simulator. Simulators such as Netsquid and SeQUeNCe have the advantage that they provide a lot of detail regarding the implementation of the physical layer as well as the network layer and the influence of quantum errors. For the field of quantum networks, there are simulators that are primarily suited for testing protocols and

> For the field of quantum networks, there are simulators that are suitable for testing protocols, for example.

creating network topologies. These include, for example, QuNetSim, SQUANCH, SimulaQron and QuISP. QuISP can even be used to simulate networks with a hundred nodes, which can also contain quantum repeaters. SQUANCH and QuNetSim are particularly suitable for creating and testing network protocols. If you want to work with QKD protocols at the application layer, you can use QKDNetSim, for example.

Simulation examples for the areas of network layer and physical layer The SQUANCH simulator is available as an open-source Python package. The software enables the creation of network topologies and the testing of protocols, but also has tools to simulate quantum technology at the circuit level. Figure 5 shows the implementation of Qubit entanglement.

In SQUANCH, communicating partners are referred to as agents. Depending on the use case, a suitable quantum circuit or logic in the form of Python code must be implemented in these agents so that they can communicate with other agents in the sense of the application. The agents themselves are connected via quantum channels to their own quantum error models, as they can occur during transmission. In the figure, two independent Qubits A and B are shown over a circuit for entanglement of the two Qubits. There the Hardamard gate (H) is first applied to a Qubit A, which is supposed to bring the Qubit in a superposition. Then over a CNOT gate, which is supposed to create the entanglement, Qubit A is connected to Qubit B.




```
class ShorQChannel(QChannel):
    '''Represents a quantum channel
    with a Shor error applied'''

    def __init__(self, from_agent, to_agent):
        QChannel.__init__(self, from_agent, to_agent)
        # register the error model
        self.errors = [ShorError(self)]
```

QError

Figure 5: Entanglement circuit and code for quantum errors in SQUANCH

After that, the entangled Qubits are distributed to the two agents (A = Alice and B = Bob). In quantum channel of B is integrated also a quantum error model. The code snippet to the error model is shown in the figure as QError.

The **QuNetSim** simulator is a package, developed in Python, that is freely available and can be used for quick and easy testing of protocols. The software simulates the network layer in a quantum network, without the user has to care about the routing between two hosts, which are (in)directly connected by the network topology are connected. In addition, the simulator has mechanisms for the control of synchronization in the network. In QuNetSim can also a classic and a quantum network are driven in parallel in a simulation. Unlike SQUANCH there is no need to define any internal circuits in the single nodes, but for example, the command "run_protocol(generate_entanglement)" is sufficient. Figure 7 shows the demonstration of a Qubit transmission, where a network of three hosts (A, B, C) defined in QuNetSim and three Qubits are transmitted from host A to C via host B.

As from the code snippet #Netzwerktopologie can be seen in Figure 7, the three hosts are defined and connected to each other corresponding to the described topology. A transmitter (host_A) and a receiver (host_B) are initialized with a protocol

```
node_1.run_protocol(generate_entanglement)
network.quantum_routing_algo = routing_algorithm
choices = ['00', '11', '10', '01']
A.send_superdense(B.host_id, random.choice(choices), await_ack=True)
```

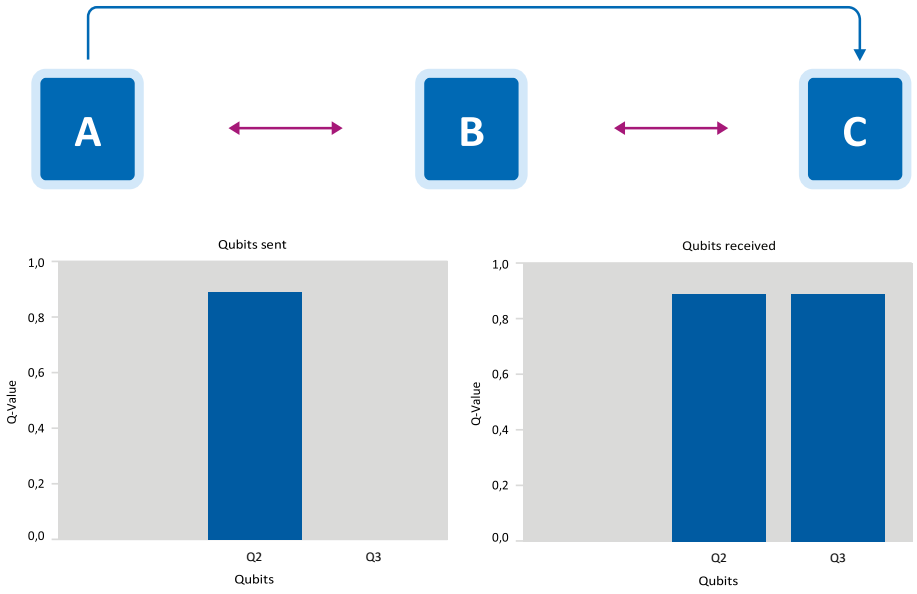
Figure 6: Making Entanglement in QuNetSim

for sending and receiving, respectively, a Qubit (#Protokolle zuweisen). In the transmitter protocol (#Protokoll Sender), three Qubits are to be transmitted to the receiver in superimposed state (Hadamard gate). As can be seen at diagrams on the right side of Figure 7, the state of the sent Qubits differs from the received Qubits: The reason for this is that after being measured, a Qubit has a 50 percent probability (Bell state) of one of the two values 0 or 1.

The **SeQUeNCe** simulator is a freely available open-source simulator that can be used for simulating influencing factors such as for example time and attenuation on the generation and transmission of Quantum States in Networks.

This allows processes to be tested in the hardware- and network layer. In the following example the Barrett-Kok-protocol [1] is presented. This is a process for generating entangled qubits, which has a high tolerance to faults such as detector failure and spontaneous emission of a photon at the hardware level. That protocol needs access to two quantum memories (Quantum Nodes) and a BSM (Bell State Measurement) node for generating

1 Barrett S D and Kok P 2005 Efficient high-fidelity quantum computation using matter qubits and linear optics Phys. Rev. A 71 060310, https://journals.aps.org/pra/abstract/10.1103/Phy sRevA.71.060310




```
#Netzwerktopologie
host_A = Host('A')
host_A.add_connection('B')
host_B=Host('B')
host_B.add_connections(['A','C'])
host_C=Host('C')
host_C.add_connection('B')


#Protokolle zuweisen
host_A.run_protocol(sender)
host_C.run_protocol(receiver)


#Protokoll Sender
def sender(host,receiver,qubits):
    for i in range(qubits):
        qubit=Qubit(host)
        qubit.H()
        host.send_qubit(qubit)
```

Figure 7: Qubit transfer with QuNetSim

entanglement and then regulates the synchronization of the entanglement process, among other things.

Unlike the two simulators already presented, SeQUeNCe is very hardware-related,

Also the function for collecting the simulation data, depending on the user-defined parameters (code snippet on the right in Figure 8), must be created by yourself. SeQUeNCe is therefore complex and requires a higher-level effort

```
# Look for free Quantum memory
eg_rule_condition(memory_info, manager,
# Actions on node 1 when condition meet
eg_rule_action1(memories_info, args):
# Actions on node 2 when condition meet
eg_rule_action2(memories_info, args):
```

```
test(sim_time, cc_delay, qc_atten, qc_dist):
"""
sim_time: Simulationsdauer (ms)
cc_delay: Verzoegerung klassischer Kanal (ns)
qc_atten: Dämpfung Quantenkanal (db/m)
qc_dist: Laenge Quantenkanaele (km)
"""
```

Figure 8: Generating Entanglement in SeQUeNCe with Code Snippets

which means that quantum physical phenomena must be taken into account when creating simulations. The code snippet on the left in Figure 8 shows the functions needed to acquire and synchronize the two quantum memories to be implemented to generate entanglement of users.

during induction. However, this simulator can be used to represent quantum networks more realistically and to better investigate and graphically represent the behavior, or the dependence on various influencing factors such as for example the attenuation.

About the simulators:
QKDSimulator: https://www.qkdsimulator.com/
SQUANCH: https://att-innovate.github.io/squanch/overview.html
QuNetSim: https://tqsd.github.io/QuNetSim/
SeQUeNCe: https://sequence-toolbox.github.io/index.html

To the platforms:
IBM Quantum: https://quantum-computing.ibm.com/
Google AI: https://quantumai.google/quantumcomputing-service Amazon Braket: https://aws.amazon.com/de/braket/ Quantum Network Explorer: https://www.quantum-network.com

Outlook

Quantum simulators are suitable for different applications and areas of application, depending on the category. Various offers of quantum platforms enable researchers to test simulations on web-based platforms and real quantum hardware (IBM) or to test the behavior and states of Qubits on supercomputers designed for this purpose (LRZ). Since quantum repeaters are still being worked on, simulators can help to create and investigate applications and protocols for networks without the hardware required for them. With the knowledge gained from this, algorithms and programs for quantum networks can be implemented even today and realized more quickly and easily when the components become available in the future.

More information on quantum research and the quantum internet can be found on **https://www.win-labor.dfn.de/.**