Blind and Distributed Quantum Computation

Evan Borras and Piper C. Wysocki

University of New Mexico

June 14, 2023

Lecture Outline

Contents

Review

- **On May 31st,** we discussed the theoretical basis and practical construction of *quantum repeaters, quantum memories, and entanglement generation*
- **On June 7th,** we discussed *quantum key distribution and other cryptographic tasks* exploiting access to trusted repeaters, prepare and measure networks, and entanglement generation

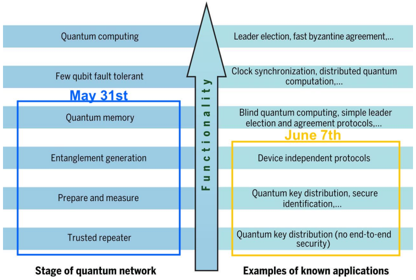

Quantum computing
Few qubit fault tolerant
May 31st
Quantum memory
Entanglement generation
Prepare and measure
Trusted repeater
Stage of quantum network
Functionality
Leader election, fast byzantine agreement,...
Clock synchronization, distributed quantum computation,...
Blind quantum computing, simple leader election and agreement protocols,...
June 7th
Device independent protocols
Quantum key distribution, secure identification,...
Quantum key distribution (no end-to-end security)
Examples of known applications

Figure: Roadmap for the quantum Internet (modified from [1])

Looking Forward

- **Today** we will focus on *distributed and blind quantum computation*, which rely on the existence of quantum memory and few qubit fault-tolerant devices

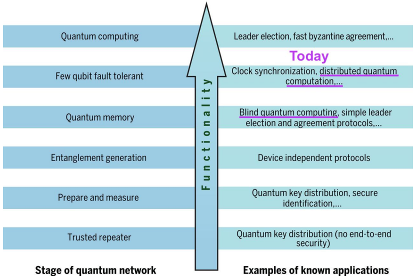
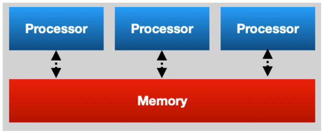

Quantum computing
Leader election, fast byzantine agreement,...
Today
Few qubit fault tolerant
Clock synchronization, distributed quantum computation,...
Quantum memory
Blind quantum computing, simple leader election and agreement protocols,...
Entanglement generation
Device independent protocols
Prepare and measure
Quantum key distribution, secure identification,...
Trusted repeater
Quantum key distribution (no end-to-end security)
Functionality
Stage of quantum network
Examples of known applications

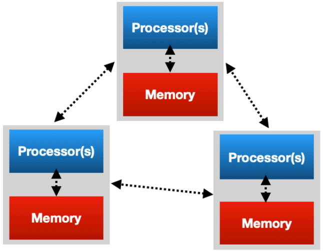
Figure: Road map for the quantum Internet (modified from [1])

Distributed and Parallel Computing

- **Parallel computing** uses *multiple processors* within a computer with access to *a single, shared memory* to solve a computational problem split into smaller operations which can be run *in parallel*, resulting in a speedup.
- **Distributed computing** uses *multiple, possibly geographical distant devices* each with *their own memories*. The computational problem is *distributed* across the available computational nodes, with information shared using communication channels.



Parallel Computation



Distributed Computation

Modular Quantum Architectures

- For each major qubit technology, there exists physical limitations on how many qubits can be put on a single quantum "chip" [2]
- Three possible solutions to continue scaling available qubits:
  - Chip-to-chip coupling to create **multi-chip quantum processors**
  - **Circuit knitting** where classical computations "knit" back together a larger circuit from smaller components distributed across quantum processors [3]
  - Clustering quantum processors using a quantum internet, where the quantum entanglement allows **distributed quantum computing**

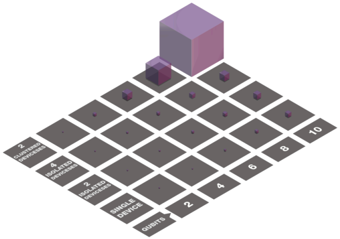
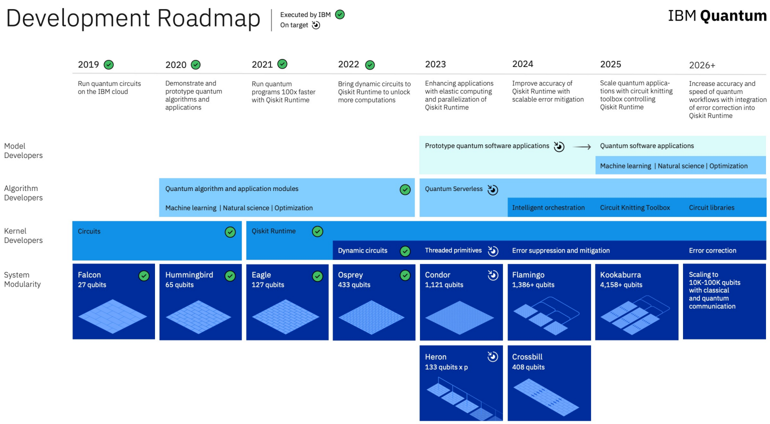


Figure: Clustering quantum processors creates an exponential speedup [2]

IBM Quantum Development Roadmap



Development Roadmap | Executed by IBM ✓ On target

IBM **Quantum**

2019 ✓	2020 ✓	2021 ✓	2022 ✓	2023	2024	2025	2026+
Run quantum circuits on the IBM cloud	Demonstrate and prototype quantum algorithms and applications	Run quantum programs 100x faster with Qiskit Runtime	Bring dynamic circuits to Qiskit Runtime to unlock more computations	Enhancing applications with elastic computing and parallelization of Qiskit Runtime	Improve accuracy of Qiskit Runtime with scalable error mitigation	Scale quantum applications with circuit knitting toolbox controlling Qiskit Runtime	Increase accuracy and speed of quantum workflows with integration of error correction into Qiskit Runtime

Model Developers: Prototype quantum software applications → Quantum software applications; Machine learning | Natural science | Optimization

Algorithm Developers: Quantum algorithm and application modules ✓; Machine learning | Natural science | Optimization; Quantum Serverless; Intelligent orchestration; Circuit Knitting Toolbox; Circuit libraries

Kernel Developers: Circuits ✓; Qiskit Runtime ✓; Dynamic circuits ✓; Threaded primitives; Error suppression and mitigation; Error correction

System Modularity:

Falcon ✓ 27 qubits	Hummingbird ✓ 65 qubits	Eagle ✓ 127 qubits	Osprey ✓ 433 qubits	Condor 1,121 qubits	Flamingo 1,386+ qubits	Kookaburra 4,158+ qubits	Scaling to 10K-100K qubits with classical and quantum communication
				Heron 133 qubits x p	Crossbill 408 qubits		

Figure: IBM Quantum Development Roadmap [3]

Modular Architecture in the IBM Roadmap

- 2023 *Heron* processors will be able to classically communicate with each other
- 2023 *Crossbill* multi-chip processors will test inter-chip coupling
- 2024 *Flamingo* processors will be capable of quantum communication between processors and quantum parallelization
- 2025 *Kookaburra* processors will allow combination of these advances to create a 4158+-qubit system

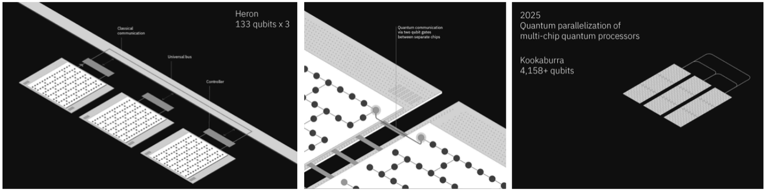

Heron
133 qubits x 3
Classical communication
Universal bus
Controller
Quantum communication via two qubit gates between separate chips
2025
Quantum parallelization of multi-chip quantum processors
Kookaburra
4,158+ qubits

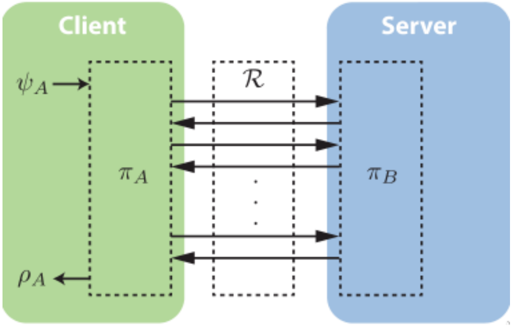
Figure: Left to right: classical communication link, quantum communication link, both communication links plus multi-chip coupling [3]

Parallel/Distributed Computing in the IBM Roadmap

- This year, **Qiskit Runtime**, IBM's computing service for workload optimization for cloud-based quantum computing, will be updated to allow parallel computation on quantum processors and automatically parallelize perfectly distributable circuits [3]
- In 2025, IBM plans to introduce a toolbox for circuit knitting, as part of their push for **quantum serverless** architecture where the user focuses on coding, not on how to optimize for the quantum hardware

Blind Computing

Blind computation is a type of cloud computing scenario where the server has none to very limited knowledge of the type of computation the client desires to server to compute [4]



Blind Computing

The ideal blind computing scenario leaks no more information to the server other than an upper bound on the time and space resources needed to perform the computation.

Abadi, Feigenbaum, and Kilian showed that classically, the client must leak more information than the space and time resources required, if the client requests the server to compute NP-hard functions [5].

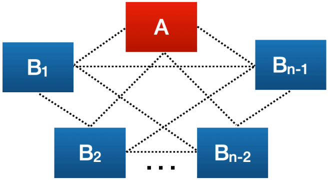
Interestingly, augmenting the client with limited QC capability or allowing multiple non-interacting QC servers allows for perfect blindness regardless of the computation [4].

Contents

Cost Comparison of Isolated and Clustered Processors [6]

Consider a quantum network with n nodes. We identify a central processor A and label the remaining nodes B_i where $i \in 1, \ldots, n-1$. We can find the cost of computation by considering the following:

- $P(n)$: The cost of precomputation required to prepare the n nodes in initial states using classical and quantum computation
- Z: The cost of running a quantum processor at a node
- Y: The cost of communicating from measurement results from a B node to the central node A
- $R(n)$: The number of times entire computation must be repeated to achieve a desired precision



Cost Comparison of Isolated and Clustered Processors [6]

- *Isolated* quantum processors whose initial states are *disentangled* do not require precomputation

$$C_1(n) = R_1(n)[nZ + (n-1)Y] \tag{1}$$

- *Clustered* quantum processors in a quantum network require precomputation due to the presence of entanglement

$$C_2(n) = R_2(n)[P_2(n) + nZ + (n-1)Y] \tag{2}$$

Cost Comparison of Isolated and Clustered Processors [6]

Consider the ratio

$$\frac{C_2(n)}{C_1(n)} = \frac{R_2(n)}{R_1(n)} \frac{P_2(n) + nZ + (n-1)Y}{nZ + (n-1)Y} \tag{3}$$

- Entanglement decreases the number of repetitions required in the absence of noise, i.e. $R_1 > R_2$. There exists some n_{min} such that for $n > n_{min}$ the number of repetitions R_2 is large enough to compensate for the precognition cost.
- $R_2(n)$ grows more rapidly in the presence of noise, since entangled states are more prone to error. There exists some n_{max} such that for $n > n_{max}$ clustering is no longer efficient.
- If $n_{min} < n_{max}$, **there exists a range $[n_{min}, n_{max}]$ where clustering processors to perform distributed quantum computation is more efficient than the same processors acting in isolation.**

Circuit Knitting: General Process

Circuit knitting is a method of **classically parallelizing quantum devices** and roughly works as follows [7], [8]:

1. Slice the larger quantum circuit into smaller, weakly-entangled pieces
2. Compute these pieces, either sequentially on the same quantum processors or in parallel using multiple processors (reduces the size of the quantum processor required!)
3. Use classical postprocessing (exponentially scaling in number of non-local gates) and sub-circuit results to simulate the larger circuit's results

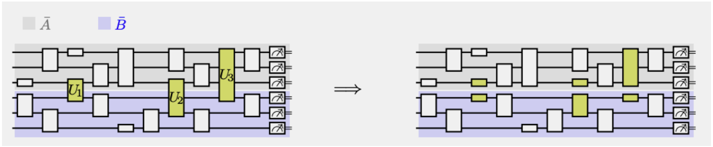


Figure: The non-local circuit (left) can be simulated by repeatedly sampling from two local circuits (right) [9]

Circuit Knitting using Quasiprobability Simulation [9]

- To simulate a non-local gate corresponding to unitary channel $\mathcal{U}$, we perform a *quasiprobability decomposition*

$$\mathcal{U} = \sum_i a_i \mathcal{F}_i \tag{4}$$

  where a_i are real coefficients (**can be negative!**) and $\mathcal{F}_i$ are the operations realizable by the hardware
- During circuit execution, the non-local gate $\mathcal{U}$ is replaced by a random $\mathcal{F}_i$ (corresponds to two circuits with local operations and any allowed classical communication)
- These sub-circuits are sampled from to allow reconstruction of the full circuit, sampling overhead scales as κ^2 where

$$\kappa = \sum_i |a_i| \tag{5}$$

Classical Communication Boosts Circuit Knitting [9]

- Joint simulation of n non-local CNOTs is considerably cheaper than simulating a non-local CNOT n-times
- For a circuit containing n non-local CNOT gates connecting two sub-parts of a larger circuit, two-way classical communication reduced the simulation overhead from $O(9^n)$ to $O(4^n)$
- Improvement also were possible for other Clifford gates and limited non-Clifford gates assuming *multiple* instances

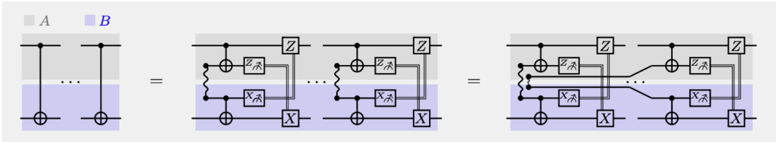


Figure: Two CNOTs (right) can be implemented using joint preparation of Bell pairs (left) instead of sequential preparation (middle) which reduces simulation overhead [9]

Even clustering quantum devices with classical communication (as with IBM *Heron* processors) improves performance!

Distributing Quantum Algorithms

- A computation is *perfectly distributable* if it does not require any *non-local gates* and can be split into autonomous parts on separate processors
- Most quantum computation sub-routines (e.g. Quantum Fourier Transform) are *not* perfectly distributable
- There exist methods to perform non-local gates and enable distribution more generally, but these methods are typically resource-intensive

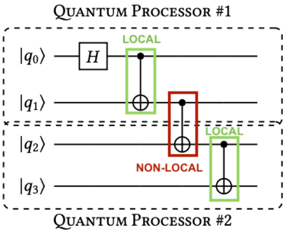
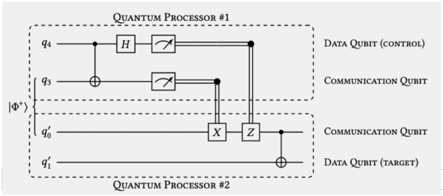


Figure: Quantum circuit distributed between two quantum processors (modified from [2])

TeleGate/TeleData

Non-local gates can be implemented in two ways:

- *TeleData* teleports the state of a data storage qubit from the first processor to a communication qubit in the second processor where the two-qubit gate is then performed locally
- *TeleGate* directly performs the gate between data qubits belonging to different processors
- Both schemes require transmission of **two classical bits**, consume **one ebit** (Bell pair), and utilize local operations



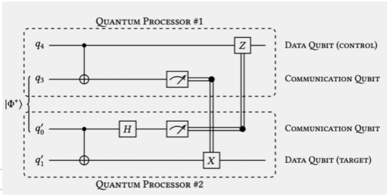


Figure: Non-local CNOT implemented using the *TeleData* (left) and *TeleGate* (right) paradigms [2]

TeleGate/TeleData: Further Notes

- Entanglement swapping allows the creation of a *virtual quantum link* enhancing connectivity and increasing the qubits between which non-local gates can be implemented
- Some papers suggest *TeleData* may perform better than *TeleGate*
    - *TeleData* is faster than *TeleGate* for quantum arithmetic algorithms [10]
    - Using both *TeleData* and *TeleGate* is faster than *TeleGate* alone for VQE and QFT (**from February!**) [11]
    - However, this is still an open question and performance of the two strategies depends on a variety of factors in the network and partition [2]

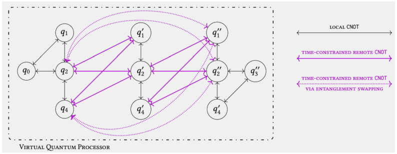
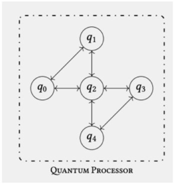

q_0
q_1
q_2
q_4
q_1'
q_2'
q_4'
q_1''
q_2''
q_3''
q_4'
LOCAL CNOT
TIME-CONSTRAINED REMOTE CNOT
TIME-CONSTRAINED REMOTE CNOT VIA ENTANGLEMENT SWAPPING
VIRTUAL QUANTUM PROCESSOR

Figure: Entanglement swapping enables additional non-local operations [2]

Quantum Compilers: Monolithic Computation

For monolithic computation, quantum compilers:

- perform *gate synthesis* to construct the required gates out of the *reduced set* able to implemented on a given quantum hardware
- map logical qubits of the quantum circuit to the physical qubits (allows fault-tolerance)
- perform *qubit routing* to make sure two-qubit gates obey the connectivity constraints between physical qubits (shown in a *coupling map*)
- **Optimization goal:** minimize compiled circuit depth (generally NP-hard)



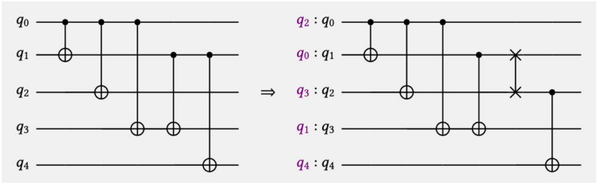


Figure: Coupling map [2]; mapping a given circuit to one obeying the coupling map [2]

Quantum Compilers: Distributed Computation

For distributed quantum computation, quantum compilers *also*:

- perform *qubit assignment*, partitioning the computation between processors
- implement *remote gate scheduling* whereby connectivity constraints between processors are considered
    - consecutive non-local gates using the same communication qubit require re-establishing an Bell pair
    - simultaneous non-local gates using the same communication qubit and a different target data qubit are impossible
- **Optimization goal:** minimize number of non-local gates/entangled bits (ebits) used in communication (generally NP-Hard)

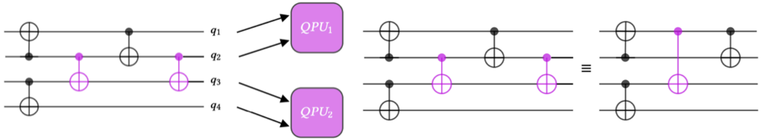


Figure: Qubit assignment [2]; ebit optimization (purple gates are non-local) [2]

Qubit Assignment as a Partitioning Problem

Qubit assignment can be represented as a partitioning problem [11], [12]

- *N* qubits become *N* vertices in an undirected weighted graph
- The weight of each edge of the graph is initially set to 0
- Each two-qubit gate between qubits q_i and q_j then adds weight 1 to the edge between vertices q_i and q_j in the graph
- **Goal:** For a *k*-way partition, minimize the total weight of inter-partition edges (generally NP-Hard to solve without heuristics [12])

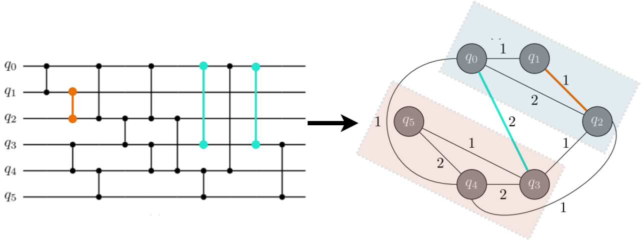


Figure: Representing a circuit (right) as a weighted graph (left) and creating two partitions (shaded boxes) (modified from [11])

Overview of the Compilation Process

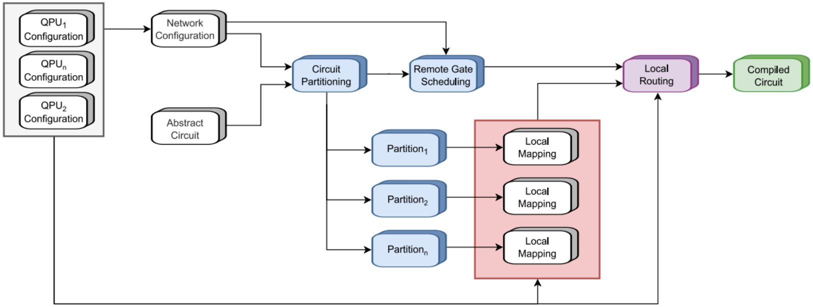


Figure: Overview of one possible compilation workflow for distributed quantum computation [11]

Discussion Question

What other considerations could you imagine would be important in distributing quantum algorithms?

Discussion Question

Possible answers: execution management, network topology, fault-tolerance

Fault-Tolerance in Distributed Quantum Computation

- Ramette *et al.* (2023) showed that surface code patches providing error correction for different modules could be connected fault-tolerantly with a threshold of $\sim 1\%$ for bulk errors and $\sim 10\%$ for interface errors (X and Z Pauli errors) [13]

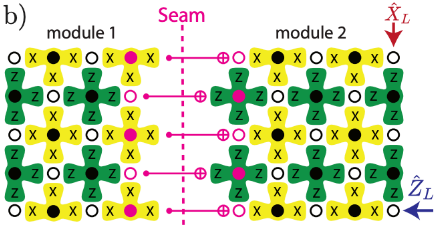


Figure: [13]

Fault-Tolerance in Distributed Quantum Computation

- Xu *et al.* (2022) showed that introducing a chip-level layer of erasure error correcting code allowed for fault-tolerance in modular systems against chip-level catastrophic errors, such as those caused by cosmic ray event (CRE) [14]

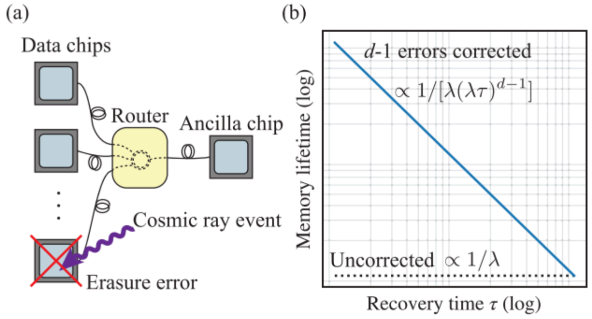
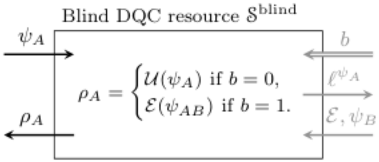


Figure: (a) Schematic of modular system with cosmic ray event; (b) Suppression of CRE errors by error correcting code [14]

Contents

Security Resource Definition of Blind QC [15]

Define: $\mathcal{S}^{blind}$ as the security resource representing the ideal blind QC scenario between client Alice (A) and server Bob (B)

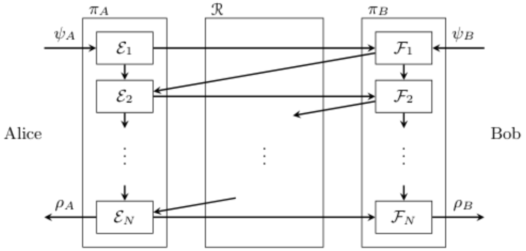

Blind DQC resource $\mathcal{S}^{\text{blind}}$
ψ_A
ρ_A
$\rho_A = \begin{cases} \mathcal{U}(\psi_A) \text{ if } b = 0, \\ \mathcal{E}(\psi_{AB}) \text{ if } b = 1. \end{cases}$
b
ℓ^{ψ_A}
$\mathcal{E}, \psi_B$

- input/output interfaces: ψ_A/ρ_A, input/output states of the client Alice's system
- filtered output interface: ℓ^{ψ_A}, the leaked space and time resources of Alice's computation
- filtered input interface: $\mathcal{E}, \psi_B$, nefarious quantum channel and auxiliary system Bob has access to if he is cheating.
- input interface: b, cheating bit, $b = 1$ gives Bob access to the filtered interfaces

Security Resource Definition of Blind QC [15]

A concrete BQC scenario involves:

- Protocol: $\pi = \{\pi_A, \pi_B\}$
  - $\pi_A = \{\mathcal{E}_1, \mathcal{E}_2, ...\}$, the protocol for Alice (also called a quantum strategy)
  - $\pi_B = \{\mathcal{F}_1, \mathcal{F}_2, ...\}$ the protocol for Bob
- Resource: $\mathcal{R}$, an ideal quantum communication channel



Symbolically we write the concrete resource as: $\pi_A \mathcal{R} \pi_B$

Security Resource Definition of Blind QC [15]

Consider a concrete implementation of BQC: $\pi_A \mathcal{R} \pi_B$

- the concrete protocol is ϵ-correct:
  - $\pi_A \mathcal{R} \pi_B \approx_\epsilon \mathcal{S}^{blind} \perp_B \iff d(\pi_A \mathcal{R} \pi_B, \mathcal{S}^{blind} \perp_B) \leq \epsilon$
  - $\perp_B$ is a filter attached to Bob's interfaces that filters out Bob's abilities to cheat (ie the resource sets $b = 0$)
  - $d(\mathcal{E}, \mathcal{F}) = \frac{1}{2} \|\mathcal{E} - \mathcal{F}\|_\diamond = \frac{1}{2} \max_\rho \{\mathrm{Tr}|\mathcal{E} \otimes \mathbb{I}(\rho) - \mathcal{F} \otimes \mathbb{I}(\rho)|\}$, since both $\mathcal{S}^{blind}$ with Bob's cheating abilities filtered out, and the concrete protocol with Bob behaving honestly form quantum channels on Alice's system
- the concrete resource is ϵ-secure
  - $\exists \sigma_B \left[\pi_A \mathcal{R} \approx_\epsilon \mathcal{S}^{blind} \sigma_B\right] \iff \exists \sigma_B \left[d\left(\pi_A \mathcal{R}, \mathcal{S}^{blind} \sigma_B\right) \leq \epsilon\right]$
  - σ_B is a simulator/converter that attaches to Bob's input/output interfaces on $\mathcal{S}^{blind}$ and converts the output from the ideal scenario towards that of the concrete scenario
  - $d(\pi_i \mathcal{R}, \pi_j \mathcal{R}) =$ the distinguishing distance between quantum strategies/combs

Distinguishing Distance between Quantum Strategies

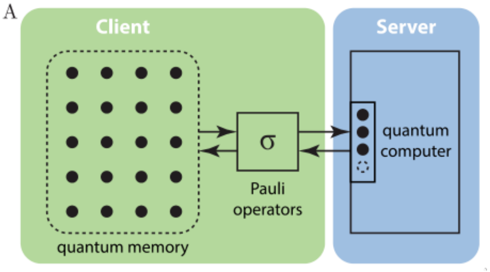
Consider the problem of distinguishing between scenarios $\pi_i\mathcal{R}$ and $\pi_j\mathcal{R}$ when given access to only one party (A) of a two player protocol.

The single shot distinguishablity distance is:

$$d(\pi_i\mathcal{R}, \pi_j\mathcal{R}) = \max_{\rho_{AR},\pi_B} \{\mathrm{Tr}|\pi_i\mathcal{R}\pi_B\left(\rho_{AR}\right) - \pi_j\mathcal{R}\pi_B\left(\rho_{AR}\right)|\}$$

see: Gutoski [16] and Chiribella et al. [17]

Child's BQC Protocol [18]



Bob's Protocol π_B:

1. On every query for computation from Alice, load $|\psi_R\rangle$ into $|\psi_B\rangle$.
2. Update/initialize U_B with the next gate from ordered set: $[H, CNOT, T, S]$ and apply U_B to $|\psi_B\rangle$
3. Load $|\psi_B\rangle$ into $|\psi_R\rangle$ and notify Alice that the computation has been performed.

Child's BQC Protocol [18]

Alice's Protocol π_A:

1. Update/initialize U_B with the next gate from ordered set: $[H, CNOT, T, S]$
2. If U_B is a gate that needs to be applied to qubit(s) i in the circuit Alice wants to compute, select qubit(s) $|\psi_A\rangle_i$ else select $|\psi_{junk}\rangle$
3. Apply random Pauli operation $\sigma(r)$ to selected qubit(s)
4. Load $|\psi_R\rangle$ with selected qubit(s), and query Bob to perform a computation
5. Alice is notified by Bob that $|\psi_R\rangle$ holds the result of his computation
6. Load $|\psi_R\rangle$ back into either $|\psi_A\rangle_i$ or $|\psi_{junk}\rangle$. If $|\psi_{junk}\rangle$ was loaded, go to the first step
7. If $U_B \in$ Clifford operations or commutes with $\sigma(r)$, decode by applying Pauli operation $\sigma' = U_B\sigma(r)U_B^\dagger$, go to the first step
8. Load $|\psi_A\rangle_i$ back into $|\psi_R\rangle$, and query Bob to perform computation. Update U_B with the next gate, and go to step 5

Child's BQC Protocol [18]

Motivation for $\epsilon = 0$ correctness:

- $U_B \in$ Clifford operations and is applied on a non-junk qubit of Alice's, decoding with $\sigma' = U_B \sigma(r) U_B^\dagger \implies |\psi_A\rangle_i \mapsto U_B |\psi_A\rangle_i$
- $U_B = T$ and is applied on a non-junk qubit of Alice's:
  - if T commutes with $\sigma(r)$, decoding with $\sigma' = T\sigma(r)T^\dagger = \sigma(r)$ implies $|\psi_A\rangle_i \mapsto \sigma(r)T\sigma(r)|\psi_A\rangle_i = T|\psi_A\rangle_i$
  - if T does not commute with $\sigma(r)$ then $T\sigma(r) = \sigma(r)T^\dagger$.
    By querying Bob, the next computation he performs is an S-gate.
    Decoding according to $\sigma' = S\sigma(r)S^\dagger$ implies
    $|\psi_A\rangle_i \mapsto \sigma' ST\sigma(r)|\psi_A\rangle_i = \sigma' S\sigma(r)T^\dagger|\psi_A\rangle_i = ST^\dagger|\psi_A\rangle_i = T|\psi_A\rangle_i$

Assuming Bob follows π_B, Alice can perform arbitrary $\{H, CNOT, T, S\}$ gates on $|\psi_A\rangle$ which is an universal gate set

Child's BQC Protocol [18]

Motivation for $\epsilon = 0$ Security:

Note: Every time Alice queries Bob to perform a computation, the state sent over to Bob is the maximally mixed state. According to Bob the state he sees is:

$$\frac{1}{4}\sum_{ij=0}^{1} Z^i X^j |\psi\rangle\langle\psi| X^j Z^i = \frac{\mathbb{I}}{2}$$

Intuitively, Bob learns nothing about the state of Alice's system since he only sees the maximally mixed state.

He also learns nothing about the function Alice wants to compute since he can not distinguish with certainty whether he is given a junk qubit or not.

Child's BQC Protocol [18]

To show $\epsilon = 0$ we need to construct some simulation using $\mathcal{S}^{blind}$ and simulator σ_B that reproduces the same outputs and inputs Alice and Bob see when $\pi_A \mathcal{R}$ is fixed.

Sketch of simulator σ_B:

- set $b = 1$
- σ_B has an output interface ψ'_B and input interface U'_B
- Each step t out of $\ell^{\psi_A}_{time}$ steps, σ_B cycles through $[H, CNOT, T, S]$ outputting the 1 or 2 qubit MM state to Bob depending on the step in the cycle.
- Each step Bob inputs a gate to perform $U'_B(t)$
- The simulator feeds the operation $\sigma'(r_1)U'_B(1)\sigma(r_1) \otimes ... \otimes \sigma'(r_{\ell^{\psi_A}_{time}})U'_B(\ell^{\psi_A}_{time})\sigma(r_{\ell^{\psi_A}_{time}})$, and system ψ_B in the state needed to teleport each $\sigma'(r_t)U'_B(t)\sigma(r_t)$ operation into $\mathcal{S}^{blind}$

Child's BQC Protocol [18]

Sketch of implementation of map $\mathcal{E}$

- S^{blind} applies the operation fed in by the simulator onto a portion of system ψ_B
- $\mathcal{S}^{blind}$ performs gate teleportation involving each qubit(s) with an operation $\sigma'(r_t)U'_B(t)\sigma(r_t)$ applied, and qubit(s) of inputted system ψ_A (which includes Alice's junk qubits)
- the order of teleportations follows the same order in which the $U'_B(t)$s were inputted
- The operations are teleported onto the states of qubits in system ψ_A, such that the correct unitary Alice wants is reproduced when $U'_B(t) = U_B(t)$ for all t
- The resulting state, after all the teleportations are finished, is swapped back onto system ψ_A
- ρ_A is outputted by tracing out all of the B subsystems, and junk qubits of the A system.

Discussion Question

Construct a mind-map (visual depiction) of the topics in Distributed QC and Blind QC and how they relate to one another.

Contents

Summary and Further Discussion

- **Distributed Quantum Computation**
  - Allows dramatic scaling qubits available for quantum computational tasks
  - Requires both classical and quantum communication (to distribute Bell pairs) between quantum processors
  - Circuit knitting serves as intermediate step, where classical postprocessing is used to "knit" together parallelized sub-circuits (classical communication provides an advantage here)
  - Considerable research remains to determine the best way to compile a distributed quantum circuit
- **Blind Quantum Computation**
  - A distributed quantum computing scenario where a client can leverage (a more powerful) quantum server(s) to perform private computations
  - Perfect privacy can be realized for quantum capable clients or classical clients with multiple servers
  - A perfect private BQC protocol with a classical client is an open question with interesting practical applications and implications (non-existence implies $BPP \neq BQP$, existence implies $NP \subseteq BQP \implies PH$ collapses to 3^{rd} level) [19]

References I

[1] S. Wehner, D. Elkouss, and R. Hanson, "Quantum internet: A vision for the road ahead," *Science*, vol. 362, no. 6412, eaam9288, Oct. 19, 2018, ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.aam9288. [Online]. Available: https://www.science.org/doi/10.1126/science.aam9288.

[2] M. Caleffi, M. Amoretti, D. Ferrari, *et al.*, *Distributed quantum computing: A survey*, Dec. 20, 2022. DOI: 10.48550/arXiv.2212.10609. arXiv: 2212.10609[quant-ph]. [Online]. Available: http://arxiv.org/abs/2212.10609.

[3] J. Gambetta, "IBM quantum roadmap to build quantum-centric supercomputers," IBM Research Blog, (Feb. 9, 2021), [Online]. Available: https://research.ibm.com/blog/ibm-quantum-roadmap-2025.

[4] J. F. Fitzsimons, "Private quantum computation: An introduction to blind quantum computing and related protocols," *npj Quantum Information*, vol. 3, no. 1, Jun. 2017. DOI: 10.1038/s41534-017-0025-3. [Online]. Available: https://doi.org/10.1038/s41534-017-0025-3.

References II

[5] M. Abadi, J. Feigenbaum, and J. Kilian, "On hiding information from an oracle," *Journal of Computer and System Sciences*, vol. 39, no. 1, pp. 21–50, 1989, ISSN: 0022-0000. DOI: https://doi.org/10.1016/0022-0000(89)90018-4. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0022000089900184.

[6] J. I. Cirac, A. K. Ekert, S. F. Huelga, and C. Macchiavello, "Distributed quantum computation over noisy channels," *Physical Review A*, vol. 59, no. 6, pp. 4249–4254, Jun. 1, 1999. DOI: 10.1103/PhysRevA.59.4249. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.59.4249.

[7] R. Davis, "Simulating large quantum circuits on small quantum computers," IBM Research Blog, (Feb. 9, 2021), [Online]. Available: https://research.ibm.com/blog/circuit-knitting-with-classical-communication.

References III

[8] D. Sutter and C. Piveteau, “IBM researchers give quantum simulation technology a boost,” IBM Research Blog, (Feb. 9, 2021), [Online]. Available: https://research.ibm.com/blog/quantum-entanglement-forging.

[9] C. Piveteau and D. Sutter, *Circuit knitting with classical communication*, Feb. 9, 2023. DOI: 10.48550/arXiv.2205.00016. arXiv: 2205.00016 [quant-ph]. [Online]. Available: http://arxiv.org/abs/2205.00016.

[10] R. Van Meter, W. J. Munro, K. Nemoto, and K. M. Itoh, “Arithmetic on a distributed-memory quantum multicomputer,” *ACM Journal on Emerging Technologies in Computing Systems*, vol. 3, no. 4, pp. 1–23, Jan. 2008, ISSN: 1550-4832, 1550-4840. DOI: 10.1145/1324177.1324179. arXiv: quant-ph/0607160. [Online]. Available: http://arxiv.org/abs/quant-ph/0607160 (visited on 06/13/2023).

References IV

[11] D. Ferrari, S. Carretta, and M. Amoretti, *A modular quantum compilation framework for distributed quantum computing*, May 4, 2023. DOI: 10.48550/arXiv.2305.02969. arXiv: 2305.02969[quant-ph]. [Online]. Available: http://arxiv.org/abs/2305.02969 (visited on 06/13/2023).

[12] O. Daei, K. Navi, and M. Zomorodi-Moghadam, "Optimized quantum circuit partitioning," *International Journal of Theoretical Physics*, vol. 59, no. 12, pp. 3804–3820, Dec. 2020, ISSN: 0020-7748, 1572-9575. DOI: 10.1007/s10773-020-04633-8. arXiv: 2005.11614[quant-ph]. [Online]. Available: http://arxiv.org/abs/2005.11614 (visited on 06/13/2023).

[13] J. Ramette, J. Sinclair, N. P. Breuckmann, and V. Vuletić, *Fault-tolerant connection of error-corrected qubits with noisy links*, Feb. 2, 2023. arXiv: 2302.01296[math-ph,physics:quant-ph]. [Online]. Available: http://arxiv.org/abs/2302.01296.

References V

[14] Q. Xu, A. Seif, H. Yan, *et al.*, "Distributed quantum error correction for chip-level catastrophic errors," *Physical Review Letters*, vol. 129, no. 24, p. 240 502, Dec. 6, 2022. DOI: 10.1103/PhysRevLett.129.240502. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevLett.129.240502.

[15] V. Dunjko, J. F. Fitzsimons, C. Portmann, and R. Renner, "Composable security of delegated quantum computation," in *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, 2014, pp. 406–425. DOI: 10.1007/978-3-662-45608-8_22. [Online]. Available: https://doi.org/10.1007/978-3-662-45608-8_22.

[16] G. Gutoski, "On a measure of distance for quantum strategies," *Journal of Mathematical Physics*, vol. 53, no. 3, Mar. 2012, 032202, ISSN: 0022-2488. DOI: 10.1063/1.3693621. eprint: https://pubs.aip.org/aip/jmp/article-pdf/doi/10.1063/1.3693621/16086429/032202_1_online.pdf. [Online]. Available: https://doi.org/10.1063/1.3693621.

References VI

[17] G. Chiribella, G. M. D'Ariano, and P. Perinotti, "Theoretical framework for quantum networks," *Phys. Rev. A*, vol. 80, p. 022 339, 2 Aug. 2009. DOI: 10.1103/PhysRevA.80.022339. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevA.80.022339.

[18] A. M. Childs, "Secure assisted quantum computation," *Quantum Info. Comput.*, vol. 5, no. 6, pp. 456–466, Sep. 2005, ISSN: 1533-7146.

[19] V. Dunjko and E. Kashefi, "Blind quantum computing with two almost identical states," *ArXiv*, vol. abs/1604.01586, 2016.