Quantum Protocols for Distributed Functional Monitoring

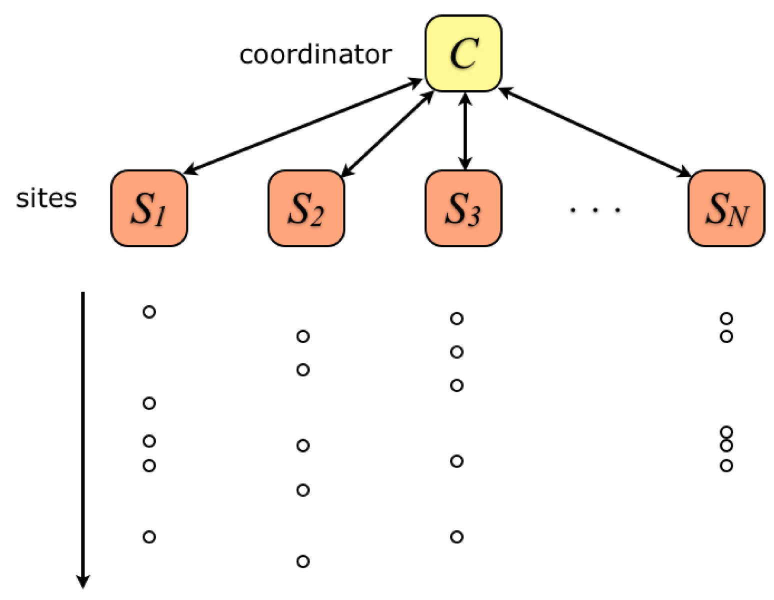
Michele Amoretti, Stefano Carretta
michele.amoretti@unipr.it, stefano.carretta@unipr.it

QIS.UNIPR.IT
QUANTUM INFORMATION SCIENCE @ UNIPR

Problem Description

In *distributed functional monitoring*, a number of players each observes a stream of items and communicates with one coordinator, whose goal is to compute a function of the union of the streams (below, a schematic of the model is illustrated). An efficient protocol is one that minimizes the communicated bits/messages between the players and the coordinator, continuously over the entire time.

In the *threshold monitoring* paradigm, the coordinator wants to know if $f(v(t)) > T$, where $v(t) = \sum_i v_i(t)/N$, being $v_i(t)$ a d-dimensional binary vector that represents the state of the stream at the i-th site ($v_i(t)$ is denoted as *stream state*), N the number of sites.



Example functions are the frequency moments $F_p = \sum_x m_x^p$, where m_x is the frequency (number of occurences) of item $x \in [n]$ (i.e., n is the cardinality of the stream alphabet). Recently, Montanaro [1] addressed the complexity of approximating the frequency moments in a centralized model where a quantum computer receives a stream of items but has only access to a limited storage space. To the best of our knowledge, no distributed functional monitoring problems have been studied in terms of quantum communication complexity.

Contribution

We consider the Geometric Monitoring (GM) method [2, 3] and we enhance it with quantum communication, thus introducing **Quantum Geometric Monitoring (QGM)**.

References

[1] Ashley Montanaro. The quantum complexity of approximating the frequency moments. *Quantum Information & Computation*, 16(13–14), 2016.

[2] Izchak Sharfman, Assaf Schuster, and Daniel Keren. A Geometric Approach to Monitoring Threshold Functions over Distributed Data Streams. In *ACM SIGMOD '06*, 2006.

[3] Nikos Giatrakos, Antonios Deligiannakis, and Minos Garofalakis. Scalable Approximate Query Tracking over Highly Distributed Data Streams. In *ACM SIGMOD '16*, 2016.

[4] Mustafa Gundogan, Patrick M. Ledingham, Kutlu Kutluer, Margherita Mazzera, and Hugues de Riedmatten. Solid state spin-wave quantum memory for time-bin qubits. *Phys. Rev. Lett.*, 114(230501), 2015.

[5] Mitali Sisodia, Abishek Shukla, and Anirban Pathak. Experimental realization of nondestructive discrimination of bell states using a five-qubit quantum computer. *arXiv:1705.00670*, 2017.

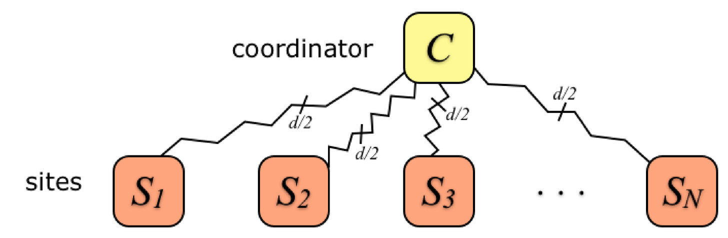
[6] Axel Dahlberg and Stephanie Wehner. SimulaQron - A simulator for developing quantum internet software. *arXiv:1712.08032*, 2017.

Quantum Geometric Monitoring (QGM)

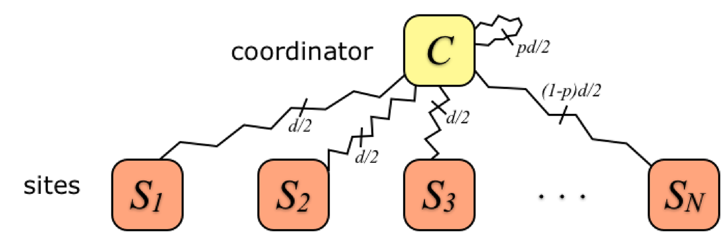
According to the GM method [2, 3], a *local violation* occurs when the locally constructed hypersphere $B(v(t) + \frac{1}{2}\Delta v_i(t), \frac{1}{2}\|\Delta v_i(t)\|)$ crosses the threshold surface. When a local violation occurs in at least one site, a synchronization step takes place where the coordinator collects the $v_i(t)$ vectors from all the N sites and assesses whether $f(v(t)) > T$. The newly computed $v(t)$ is then communicated back to the N sites. Communication cost for each synchronization period: $2dN$.

In QGM, the coordinator and each site share $d/2$ Bell pairs, whose qubits are stored in quantum memories. Each Bell pair corresponds to two bits of information (e.g., $|\beta_{00}\rangle$ corresponds to 00).

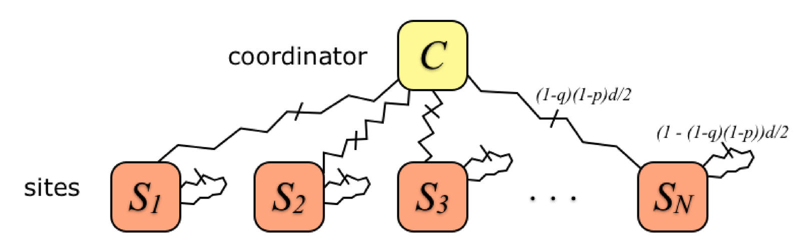
1. Initialization: the coordinator owns all Bell pairs, sets their state to $|\beta_{00}\rangle$ and sends one qubit of each pair to the sites ($d/2$ qubits per site). Communication cost: $\frac{d}{2}N$ (1)



2. When a local violation is signaled by one site, the coordinator starts collecting $v_i(t)$, for each $i \in [N]$, one site after another. This means that each site sends $p\frac{d}{2}$ qubits to the coordinator, where p is the modification probability of the shared Bell pairs, with respect to previous synchronization. Actually, photons are sent one after another (slotted communication), and their arrival order is used by the coordinator to correctly associate them to the appropriate qubits already stored in the local memory [4]. Communication cost: $p\frac{d}{2}N$ (2)



3. Once all the updated $v_i(t)$s have been received, the coordinator performs Bell state discrimination by nondestructive measurement on the shared Bell pairs (at least on those that have changed) [5]. Then, the coordinator computes the new $v(t)$ and uses the shared Bell pairs to provide it to all the sites. Let q be the modification probability of the shared Bell pairs (for $v(t)$ with respect to $v_i(t)$ for each $i \in [N]$). Communication cost: $(2p + q(1-p))\frac{d}{2}N$ (3)



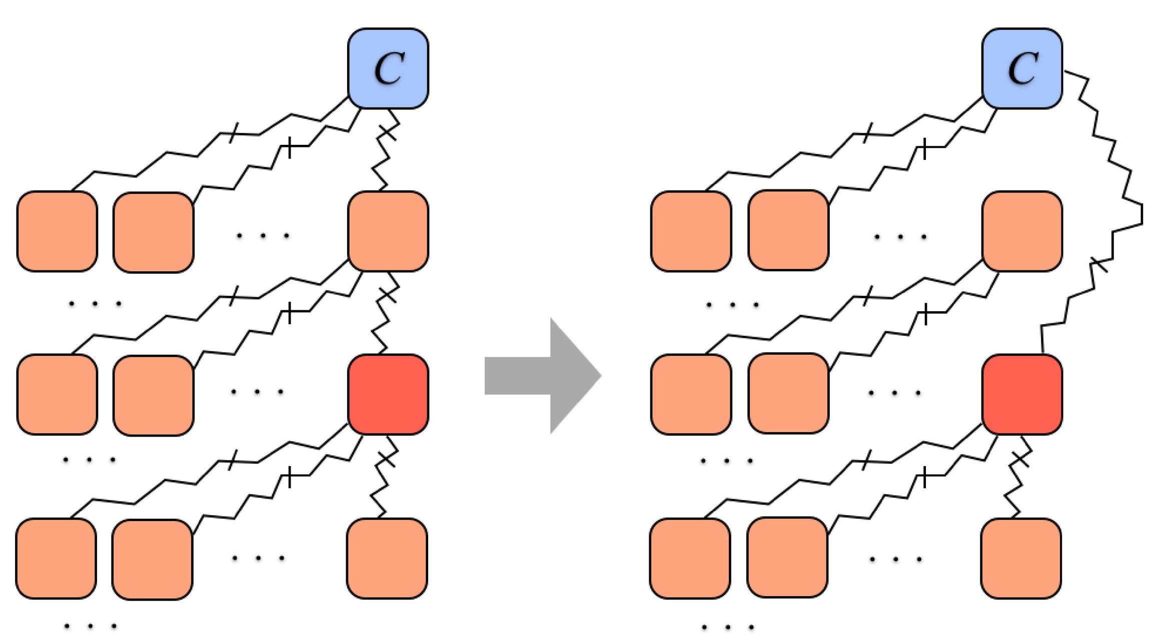
4. Upon receiving all the qubits, the sites perform Bell state discrimination by nondestructive measurement on the shared Bell pairs. Then, they re-initialize the Bell pairs, sending back some qubits to the coordinator. Communication cost: $(1-(1-q)(1-p))\frac{d}{2}N$ (4)

Steps 2 to 4 are repeated continuously. Total steady state communication cost:

$$(2) + (3) + (4) = (2p + q(1-p))dN$$

Examples: a) with $p = q = 1/2$, the communication cost is $\frac{5}{4}dN$; b) with $p = q = 1/8$, the communication cost is $\frac{23}{64}dN < \frac{d}{2}N$.

Implementation with SimulaQron



For a scalable implementation, we propose to organize the sites in a tree structure (each site having at most $M \ll N$ children), where the QGM protocol is executed in each subtree. Global monitoring is achieved by means of a quantum protocol we denote as **QGM-Tree**. When a site detects a subtree-level violation, it starts an entanglement swapping process to directly share its Bell states with the coordinator, that collects the $v_i(t)$s from the $M-1$ other children, to update $v(t)$.

We are implementing QGM and QGM-Tree with SimulaQron [6], a novel tool enabling application development and exploring software engineering practices for quantum networking.

Source code: `https://github.com/qis-unipr/QuantumNetworking`

Acknowledgements

This work has been supported by the University of Parma Research Fund - FIL 2016 - Project "NEXTALGO: Efficient Algorithms for Next-Generation Distributed Systems".