Quantum Algorithms for Data Streams

Wim van Dam

Departments of Computer Science and Physics
University of California, Santa Barbara

NASA 2012 QFT 1.0: First NASA Quantum Future Technologies Conference, NASA Ames Research Center, Moffett Field, US-CA

Thursday, January 19, 2012

This research is supported by
the National Science Foundation

Joint work with Qingqing Yuan (UCSB)

Data Streams

In the *data stream model* we have to process input that arrives sequentially and that is too large to be stored by the computer.

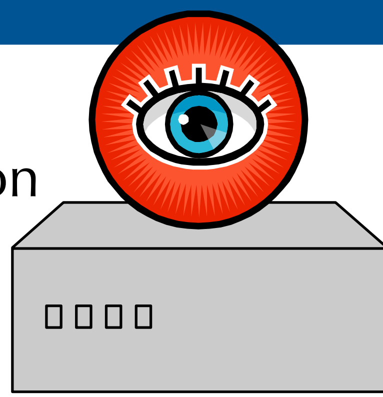
Cf. traditional setting where the input size N is 'small' and we can read and write the data repeatedly, and the time/space requirements are hopefully poly(N).

Data Stream Algorithms

A realistic setting when dealing with large streams of data in an online setting (like internet routers):

$X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_8, X_9, X_{10}, X_{11}, X_{12}, X_{13}, X_{14}, X$

We want to calculate a function on the input stream $X_1 \ldots X_N$; N is large, and the stream will only pass by once…
How much internal memory will we need?



Classic example: $X_j \in A$ with large alphabet A; what is the most frequent element in $X_1 \ldots X_N$?

Most Frequent Element

[Alon-Matias-Szegedy'99] "For sequences $X \in \{1,\ldots,M\}^N$, determining the most frequent element (approximately) requires $\Omega(M)$ bits of memory."

Worst case setting occurs when $M>N$ and almost all elements $X_1,\ldots,X_N$ are unique such that we have to keep track the frequencies f_j for all values $j \in \{1,\ldots,M\}$.

Think of routers monitoring IP addresses.

The proof uses bounds from communication complexity.

Can we do better "quantumly"?

"The Moment is Now"

Quantum Algorithms for Data Streams?

We live in a time of…

- exceptionally large data streams
- exceptionally small quantum computers

Less Quantum Memory?

Quantum algorithms on data streams:
Idea and Hope: using quantum memory to significantly reduce the memory requirements for the data stream tasks.

Arguments Con: By Holevo's theorem we know that qubits do not carry more information than classical ones.

Arguments Pro: We know that we can save memory requirements for communication complexity and for finite automata computations.

[Le Gall'06]: First exponential quantum-classical memory reduction for a specific data stream problem.

A Quantum Algorithm for Most Frequent Item Problem?

For a stream $X \in \{1,\ldots,M\}^N$ find the largest one among the M frequencies $f_j = |\{\, 1 \leqslant i \leqslant N : X_i = j \,\}|$.

Quantumly, in one pass, we can create superpositions like these $\displaystyle\sum_{j \in \{1,\ldots,M\}} |j, f_j\rangle / \sqrt{M}$

Does this give a quantum algorithm with memory requirements that are less than O(M)?

Quantum Lower Bound I

Result: the $\Omega(M)$ bound also holds in the quantum case.

Sketch of proof (rephrasing it as the Disjointness problem in communication complexity):
- Assume an algorithm with quantum memory size s.

- Assume frequencies f_j that are 0, 1 or 2.

- Let two parties A and B have 2 strings $\in \{0,1\}^M$; viewed as characteristic vectors, A and B have 2 subsets $\subseteq\{1,\ldots,M\}$; concatenate these subsets to one string X.

- Disjointness problem for $\{0,1\}^M$ is solved answering: "Is there an element with frequency > 1 in the sequence X?"

Quantum Lower Bound II

Result: the $\Omega(M)$ bound also holds in the quantum case.

Sketch of proof (rephrasing it as the Disjointness problem in communication complexity; assume s qubits of memory):

- Disjointness problem for $\{0,1\}^M$ is solved answering: “Is there an element with frequency > 1 in the sequence X?”

- A runs the data stream algorithm on the first part of X, then sends her s qubits to B, who then finishes the protocol.

- By the quantum one-way Disjointness bound: $s \in \Omega(M)$.

Repeated Inputs

We did not get a quantum improvement because we had to consider the 1-way communication complexity of Disjoint_M.

To get the quantum improvement we have to consider the data stream equivalent of multi-round communication.

This translates into assuming repeats XX…X of the input X.

Result: Given $X \in \{1,\ldots,M\}^N$, on input X^k with $k=\sqrt{M}$ there exists a quantum algorithm with $O(\log M+\log N)$ qubits that solves the most frequent element problem. Classically one needs $\Omega(\sqrt{M})$ bits of memory.

The Quantum Advantage

For $X \in \{1,\ldots,M\}^N$, let $f_1,\ldots,f_M$ denoted the frequencies.

- Parsing the string X once, the quantum algorithm can create the superposition $(\sum_j |j,f_j\rangle)/\sqrt{M}$ of log M+log N bits.

- A quantum algorithm can find the maximum frequency f_j in $\sqrt{M}$ queries, hence after $\sqrt{M}$ parsings of X the quantum algorithm knows the most frequent element.

Classically, we can use the $\Omega(M)$ lower bound for the Disjointness problem for multi-round communication to show that the memory needs to be $\Omega(\sqrt{M})$ bits for any classical data stream algorithm for the same problem.

Another Result

This one is for Cris

Let the data stream be $X=g_1,\ldots,g_N$ with all $g_j \in G$ of a (possibly non-Abelian) group G.

Identity problem: Is the product $g_1 \cdot \ldots \cdot g_N$ the identity?
Obvious solution: Keep track of the product as you see the elements pass by; this gives a $O(\log|G|)$ upper bound.

More fancy quantum solution using representation theory:
Let d_λ be the dimensions of G's irreducible representations.
There is a probabilistic quantum algorithm that on average uses $\sum_\lambda d_\lambda^2 \cdot \log d_\lambda / |G|$ qubits of memory.
For groups like $\mathbb{Z}/N\mathbb{Z}$ and D_N this implies $O(1)$ qubits.

Classically, one needs $\Omega(\log|G|)$ bits [Ambainis'98].

The Quantum Algorithm

- Before starting to read the string, pick an irreducible representation λ of G with probability $d_\lambda^2/|G|$.

- Keep track of the representation $\lambda(g_1 g_2 \ldots) \in SU(d_\lambda)$ of the product $g_1 g_2 \ldots$ using 2 log d_λ qubits as

$$(\lambda(g_1 g_2 \cdots) \otimes I)(|1,1,\rangle + \cdots + |d_\lambda, d_\lambda\rangle)/\sqrt{d_\lambda}$$

- After processing the whole string measure if the representation $\lambda(g_1 \cdot \ldots \cdot g_N)$ is the identity, or not.

- We will detect this with probability 1/2.

- Use several representations to improve success rate.

Diagram that Explains it All