Staggered Quantum Algorithm for Element Distinctness

Alexandre S. Abreu¹, Luis A.B. Kowada², Franklin L. Marquezino¹
Programa de Engenharia de Sistemas e Computação, Coppe - UFRJ¹
Instituto de Computação - UFF²
(santiago, franklin)@cos.ufrj.br, luis@ic.uff.br²

Abstract

The element distinctness problem consists in finding, given a list of elements, if there is some collision (two or more equal elements). The quantum algorithm for element distinctness was first proposed by Ambainis and uses important concepts of quantum walks. Ambainis’ algorithm performs a quantum walk over the edges of a Johnson graph while searches for a marked vertex. Szegedy generalized this process into a quantum walk model on bipartite graphs, leading to the formulation of other important algorithms. Recently, Portugal et al. introduced a staggered quantum walk, a novel quantum walk model which includes Szegedy’s model as a particular case. In this work, we reformulate the element distinctness quantum algorithm using a staggered quantum walk. We show that our new formulation requires less memory than the original one while keeping the same query complexity.

1. Introduction

The problem: In a given list of indices \( |N| = 0, 1, \ldots, N-1 \), where each index represents an element in \( |M| \), we have the following considerations:

1. **Element Distinctness.** Given elements \( v_0, v_1, \ldots, v_{N-1} \in |M| \), there are indices \( i, j \in |N| \) such that \( v_i = v_j \).
2. **Element k-Distinctness.** Given elements \( v_0, v_1, \ldots, v_{N-1} \in |M| \), there are indices \( i_1, i_2, \ldots, i_k \in |N| \) such that \( v_i = v_{i+k}, v_{i+1} \neq v_{i+k}, v_{i+2} \neq v_{i+k}, \ldots, v_{i+k-1} \neq v_{i+k} \).

**Quantum Walks (QW):** Describes the movement of a particle through nodes of a graph [2]. This walk can be reversible and deterministic, as ruled by the postulates of quantum mechanics. Two models are particularly important for this work: Szegedy’s model [7] and Staggered model [6].

Ambainis’ algorithm for element distinctness: Uses concepts of OW and quantum search in graphs to perform a search for a marked vertex on a Johnson graph, requiring \( O(\log N \cdot \log M) \) qubits of memory [5], and \( O(\sqrt{N^{k-1}}) \) queries, where \( k \) is the number of collisions in the list.

Belovs’ algorithm for element distinctness: This quantum algorithm is based in learning graphs and requires \( O(\sqrt{2^k N^{k-1}}) \) queries, being more efficient than Ambainis’ algorithm [4].

The staggered quantum algorithm for element distinctness: Our proposal uses the Staggered quantum walk model [6]. This algorithm has the same complexity as Ambainis’ original proposal, however it has the advantage of requiring just \( O(\log N) \) qubits of memory [1].

2. Staggered Quantum Walk Model

**Definition 1.** Let \( C \) be a set of cliques of \( G \). We say that \( C \) is a set of disjoint maximal cliques of \( G \) if the cliques of the set \( C \) are pairwise disjoint and the inclusion of any vertex \( v \in \bigcup_{C \in C} C \) to some clique \( C \in C \) turns it into a non-complete graph—that is, \( K_{|C|} \backslash v \) is not a clique of \( G \)—or because the cliques of \( C \) are no longer pairwise disjoint—that is, \( K_{|C|} \backslash v \) is a complete intersection of maximal cliques. The union of all Cliques \( C \) of \( G \) is a disjoint maximal set of cliques of \( G \), then we say that \( G \) is a tessellation of \( G \). In the context of the Staggered Quantum Walk model, we call each clique in \( C \) a polygon.

**Definition 2.** Let \( C_1, C_2, \ldots, C_k \) be tessellations of \( G \). We say that \( G \) is \( t \)-tessellable if and only if \( C_1 \cup C_2 \cup \cdots \cup C_k \) is the smallest family of tessellations in \( |N| \) such that \( C_1 \cup C_2 \cup \cdots \cup C_k \) covers all vertices and all edges of \( G \).

**Figure 1:** Different tessellations of the same graph. Each tessellation is a set of disjoint-maximal cliques in the graph. Notice that in each tessellation, every vertex is covered by a polygon.

**Figure 2:** A graph \( G \) covered by tessellations present in Fig 1. This graph is \( 3 \)-tessellable because if we remove any tessellation, the graph will not be fully covered.

**Figure 3:** Graph \( G \) generated from \( |N| = \{0, 1, 2, 3\} \), where \( k = 2, N = 4 \) and \( t = 2 \). We have \( a \)-tessellation in red and \( b \)-tessellation in blue.

**Algorithm 1:** Element \( k \)-distinction Algorithm (Single-Query)

1. Generate the uniform superposition \( \sum_{|S|} |S\rangle |y\rangle \).
2. Apply \( O(\log N \cdot \log M) \) times repeat:
   - (i) Apply the conditional phase flip \( (|S\rangle y\rangle) \rightarrow (|S\rangle -|y\rangle y\rangle \), such that \( v_1 = \ldots = v_t \) the \( k \)-tessellation is defined by polygons that cover cliques, where for every pair of vertices \( v, v' \) such that \( v = (v_1, y_1) \) and \( v' = (v_2, y_2) \), we have \( S = S' \) and \( y \neq y' \).
   - (ii) Apply \( (|S\rangle y\rangle) \rightarrow (|S\rangle -|y\rangle y\rangle) \) times repeat:
     - Measure the final state. Check if \( S \) contains a \( k \)-collision and answer "There is a \( k \)-collision" or "There is no \( k \)-collision" according to the result.

**References**


