DECOMPOSITION OF QUANTUM GATE INTO MULTIPLICATION OF GATES TAKEN FROM A GIVEN SET

SEDRI OGUR AND STEFANO BERNAGOZZI

Abstract. Decompose a quantum operation into a finite set of gates is extremely important in quantum information because it allows you to reproduce the operation in your laboratory even if you don't have that special gate. Here, we use linear algebra for find the decomposition of such matrices given a finite set and the starting matrix. the approach is for 4x4 matrices with 2x2 or 4x4 gates but can be generalized.

1. Introduction

Quantum computation is very useful for finding a solution to problems that on classical computer are nowadays intractable and there are many researchers trying to make quantum computation works on their small laboratories. Therefore is very important to reproduce any gate even if you don't have it. there are many articles about the decomposition of quantum gates, but here we want to find a sequence of matrices from a given set for finding the solution, knowing that the matrix is composed by multiplying that matrices. This article is composed into the following sections, section number two gives a brief explanation of quantum computation, section number three explain how is made the algorithm, section number four shows the results of the algorithm for Mathematica code and section number five gives some information about future work based on this article.

2. base results

It is well known that quantum mechanics is based on complex linear algebra, it means that all n-qubit quantum gates can be represented as 2^n by 2^n Hermitian matrices. We need those type of matrices because each quantum gate must be reversible and leave the qubits in a pure state (that means $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle\, and |\alpha|^2 + |\beta|^2 = 1$). For more informations about quantum representation see Nielsen and Chuang book[?] or Kaye, Laflamme and Mosca book [?]. By this we obtain that a composition of two gates can be written as the multiplication of the matrices of that gates, so with one matrix we can represent multiple gates.

3. algorithm

Here is the explanation of the algorithm. In section 3.1 we're going to explain how the algorithm works, in section 3.2 there is an explanation of the complexity of the algorithm with a brief introduction of the iterative deepng search algorithm and in

Date: May 11, 2015.

section 3.3 there is the pseudo-code of the algorithm. For the code of the algorithm in Wolfram Mathematica see https://github.com/ste93/matrixdecomposition.

3.1. **How it works.**

The algorithm that we propose here begin with the function that controls the matrix from the set given and if there is an U that is of the form 2 by 2 then it will replaced by two 4 by 4 matrices formed as following: U1 = U X I, U2 I x U. After this we have two matrices, U1 that acts U on the first qubit and U2 that acts U on the second qubit. After it uses an Iterative Deepening Search where at each node of the tree the branch factor is proportional to the number of the elements of the final set with all 4 by 4 matrices.

3.2. **Complexity and IDS.**

In fact the complexity of the algorithm is $O(b^d)$ where b is the branching factor of the tree (the number of the elements in the set), while d is the number of matrices needed for the solution. The completeness of the algorithm is guaranteed because we know that the matrix is a multiplication of the matrices in the set so the algorithm will perform an iterative deepening search that is know that is complete (for references see). Also the optimality of the solution is guaranteed because the iterative deepening search add at every level of the tree one matrix and explore all that level until it find a solution or reach the end of the level, so if the solution is find must be optimal.

Here there is an example of how the iterative deepening search algorithm works

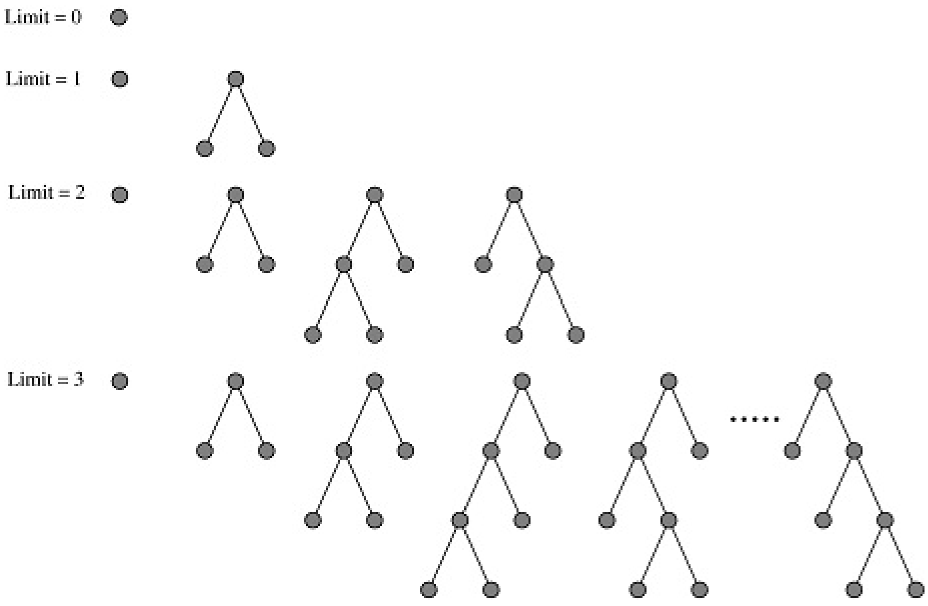

Image taken from http://www.massey.ac.nz/ mjjohnso/notes/59302/l03.html

3.3. Pseudo-code of the algorithm.

Here is the pseudo code of the algorithm:

```
def find_Matrix(set, originalMatrix, Stack, LastMatrix, depth, limit):
  if LastMatrix == originalMatrix:
    return Stack
  else:
    if depth < limit:
      for mat in set:
        if originalMatrix * mat != I4:
          find_Matrix(set, originalMatrix, Stack + mat, LastMatrix * mat, depth +


def matrixDecomposition(U, set):
  for mat in set:
    if size(mat) == 2x2:
      set.remove(mat)
      set.add(kronecker(I2,U))
      set.add(kronecker(U,I2))
  i = 0
  do:
    i++
    ris = find_matrix(set, U, stack(), I4, 0, i)
  while ris == {}
  print ris
```

Where:

1. stack() is an empty stack;
2. kronecker(A,B) is the kronecker product between A and B
3. I2 is the 2x2 identity matrix
4. I4 is the 4x4 identity matrix

4. FUTURE WORK

We will now explain the future work based on this results. At first there is maybe an improvement of the efficiency in time and space of the algorithm, as well as the generalization for matrix not 4 by 4 and see if it's possible to find the reduction of all 4 by 4 quantum gates into elements of a given set without knowing if they are or not made by multiplying elements of the set (this can be an important result because can lead to universality of a certain set). For the universality of quantum gates see Schmassman presentation [?] or Deutsch, Barenco and Ekert paper [?]. Another important work is to code the algorithm for quantum computers and see the complexity difference between them and how quantum computers affect the computation.

Last work can be the analysis of the algorithm and see if it can be reduced to NP-complete problems and if it is NP-hard.

References

[1] http://qudev.ethz.ch/content/courses/QSIT07/presentations/Schmassmann.pdf.

[2] Deutsch, David, Adriano Barenco, and Artur Ekert.: *Universality in quantum computation.*, Proceedings of the Royal Society of London. Series A: Mathematical and Physical Sciences, **449.1937**, (1995): 669-677.

[3] Nielsen, Michael A., and Isaac L. Chuang.: *Quantum computation and quantum information*, Cambridge university press, 2010.

[4] Kaye, Phillip, Raymond Laflamme, and Michele Mosca.: *An introduction to quantum computing*, Oxford University Press, 2007.

[5] https://github.com/ste93/matrixdecomposition.