Письма в ЭЧАЯ. 2007. Т. 4, № 2(138). С. 189–194

SIMULATION OF n-QUBIT QUANTUM SYSTEMS: A COMPUTER-ALGEBRAIC APPROACH

T. Radtke[1], *S. Fritzsche*, *A. Surzhykov*

Institut für Physik, Universität Kassel, Kassel, Germany

During the last decade, the field of quantum computation has attracted a lot of interest and motivated many theoretical and experimental studies of n-qubit quantum systems. But apart from the promise of more efficient quantum algorithms, these investigations also revealed a number of obstacles which still have to be overcome in practice. In this context, the use of simulation programs has proved to be an appropriate method. In order to facilitate the simulation of n-qubit quantum systems, we present the `Feynman` program to provide necessary tools to define and to deal with quantum registers as well as the operators acting on them. Using an interactive design within the framework of the computer algebra system `Maple`, we hope that the `Feynman` program will be useful not only for teaching the basic elements of quantum computing but also for studying their physical realization in the future.

С середины 1990-х гг. в теории квантовых компьютеров и квантовых вычислений был достигнут значительный прогресс. Однако физическая реализация эффективных квантовых алгоритмов требует преодоления многих технологических трудностей. Для анализа этих трудностей и, следовательно, для моделирования (физических) квантовых компьютеров в течение последних лет было разработано несколько компьютерных программ. В данной статье мы представляем программу «Фейнман», созданную для исследования системы из произвольного числа кубитов. Программа состоит из набора процедур пакета `Maple`, позволяющих симулировать эволюцию одного или нескольких квантовых регистров посредством базовых квантовых операций.

INTRODUCTION

Since Shor's discovery of an efficient factoring algorithm [1], the theory of quantum computation and quantum information has attracted a lot of interest. Among others, particularly Shor's algorithm has demonstrated that quantum computers may perform certain useful tasks much more efficiently than their classical counterparts. Despite the promising perspectives, however, there are still many practical difficulties most of which are closely related to undesirable interactions within n-qubit quantum systems or to the unwanted coupling of such systems with their environment.

Although the basic theoretical concepts of quantum computing are now well understood, their experimental realization is likely impossible without the (dynamical) behaviour of n-qubit quantum systems, so-called quantum registers, which can be simulated and analyzed in detail. To this end, a number of simulation programs have been developed in recent years which all

[1]E-mail: tradtke@physik.uni-kassel.de

provide the underlying (linear-algebra) operations in a form appropriate for general n-qubit quantum registers. A list of such quantum computer simulation programs can be found, for instance, in Refs. [2,3]. Often, however, these programs were designed for just a particular task, such as the demonstration of the superposition concept or the evaluation of (specific) quantum circuits and, hence, cannot be used for more general applications. Moreover, many of the programs are restricted to the *unitary* evolution of quantum registers being in a pure state, without the crucial possibility to take into account mixed states [4] or the coupling of the quantum registers with their environment (quantum operations) [5,6].

To facilitate the simulation of such general n-qubit quantum systems, here we present the `Feynman` program which provides necessary tools in order to deal with quantum registers and quantum operations. In contrast to most of the traditional programs [2,3], we have designed and implemented the `Feynman` code within the framework of `Maple` in order to take advantage of the various symbolic and numerical features of modern computer algebra. In the first version of our program, we provide a set of procedures to define quantum registers of variable size and to manipulate them by time-independent operators. Since a large number of such quantum operators have been predefined in the code, we expect the `Feynman` program to be useful for quite different applications, both in education and research.

1. THE `Feynman` PROGRAM

The `Feynman` program has been designed to support the simulation of n-qubit quantum systems (quantum registers). Apart from the definition and initialization of quantum registers, such a simulation requires one to transform their state either by unitary or non-unitary operations until the result of the computations (i.e. the final state) can be measured. Generally, of course, a full simulation would have to include both the desired and undesired interactions of the system with its environment. Obviously, however, not all of these requirements can be realized during the first implementation of a program. Therefore, in the first version of the `Feynman` package, our aim is to establish the basic data structures and to provide simple access to the unitary transformation of n-qubit quantum registers with no further restriction on n other than that given by the memory and time-limitations of the computer. Although the program is currently mainly focused on *unitary* transformations of quantum registers, it equally supports the representation of their states either in terms of state vectors or density matrices, including support for the concept of the reduced density matrix. A further advantage is that the `Feynman` program provides an interactive and user-friendly tool for which the knowledge of only a few (main) procedures is enough to carry out most of the computations. When compared to purely numerical implementations using traditional programming languages like `C` or `Fortran`, the `Feynman` program enables the user to perform the computation either in a symbolic or numerical form. Finally, since `Maple` by itself offers numerous built-in mathematical functions, this may help to extend the program to cover additional applications, such as quantum measures, decoherence models, error correction, and several others.

Following `Maple`'s philosophy, the `Feynman` program is organized as a hierarchy of currently about 35 procedures at quite different levels of complexity. Apart from several low-level subprocedures, which typically remain hidden to the user, the main commands can be used either for interactive work or simply as language elements in order to build new commands at some higher level of the hierarchy. Moreover, the procedures in the program

Table 1. **Selected important auxiliary procedures of the `Feynman` program to represent the basic data structures**

Procedure	Explanation
cbs()	Represents a computational basis state $\|k\rangle_{\text{dec}}$ of an n-qubit quantum register in the decimal basis $\|0\rangle, \|1\rangle, \ldots, \|k\rangle, \ldots, \|2^n - 1\rangle$
qbit()	Represents a one-qubit state $\|\psi\rangle = a\|0\rangle + b\|1\rangle$ in terms of two (complex) coefficients a and b in the computational basis $\|0\rangle \equiv \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $\|1\rangle \equiv \begin{pmatrix} 0 \\ 1 \end{pmatrix}$, respectively
qregister()	Represents an n-qubit quantum register either in terms of its 2^n dimensional state vector or the $2^n \times 2^n$ density matrix

Table 2. **Selected main commands of the `Feynman`**

Procedure	Explanation
Feynman_apply()	Applies a given qoperator() or qoperation() to the state vector or density matrix of an n-qubit quantum register
Feynman_plot_Bloch_vector()	Returns a 3D plot of the Bloch-sphere representation for a single qbit() or for the selected qubit within the given qregister()
Feynman_set_qregister()	Returns a qregister() in some (pre-defined) state such as the computational basis states, the Bell states, or several others
Feynman_trace()	Calculates the reduced density operators of a qregister() (i.e. the partial trace) and the expectation values for given matrix operators

package are divided into two groups: on the one hand, the *auxiliary* procedures which represent the logical building blocks (and data structures) of the program and, on the other hand, the *main* commands which operate on these structures. A list of the most important auxiliary procedures and some selected main procedures are displayed in Tables 1 and 2, respectively. For a complete description of the program we refer the reader to the manual which is distributed together with the program. The first version of the `Feynman` program will be published in the `CPC` library [7].

2. EXAMPLES

In order to give a short illustration of the interactive use of the `Feynman` procedures, two simple examples are displayed below as they might occur, for instance, in an introductory course on quantum computation.

2.1. The Partial Trace of a Bell State. In our first example, we would like to demonstrate how the partial trace operation is applied to a composite state. In order to examine the subsystem A of a bipartite system AB, we need a way to describe (uniquely) the system of interest while disregarding the other subsystem, B. This is achieved by the reduced density operator of A which is calculated by «tracing out» the subsystem B [6]. For an entangled state, which is by definition not a product state, the result is typically not obvious so that we have to calculate the partial trace explicitly. Consider, for instance, the Bell state $|\Phi^+\rangle = \dfrac{|00\rangle + |11\rangle}{\sqrt{2}}$. Within the Feynman program, this state can conveniently be defined by the following command:

```
> Phi := Feynman_set_qregister("Bell","Phi+");
                                             [ 1/2]
                                             [2   ]
                                             [----]
                                             [ 2  ]
                                             [    ]
                                             [ 0  ]
                  Phi := qregister(Bell, 2, [    ])
                                             [ 0  ]
                                             [    ]
                                             [ 1/2]
                                             [2   ]
                                             [----]
                                             [ 2  ]
```

After having defined a quantum register (qregister) in the Bell state $|\Phi^+\rangle$ we call the command Feynman_trace and specify the input register (Phi) and the list of qubit numbers that shall be traced out, in this case No. 1.

```
> B := Feynman_trace(Phi,[1]);
                                          [1/2     0 ]
                   B := qregister(id_, 1, [          ])
                                          [ 0     1/2]
```

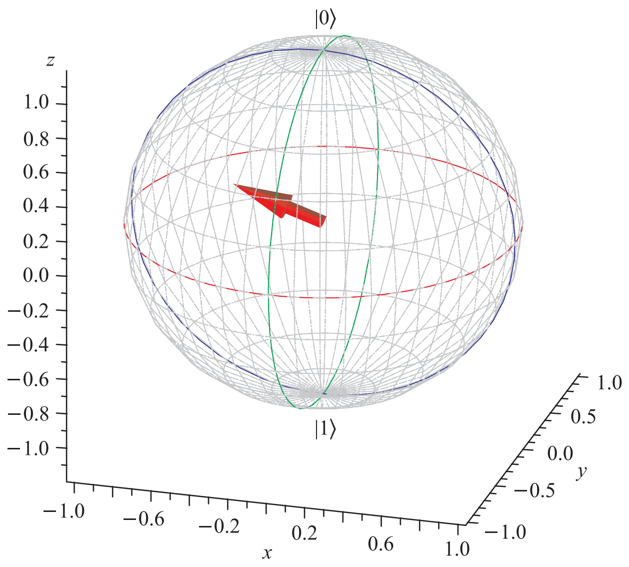
As a result, a new qregister structure is obtained which contains the reduced density matrix ρ^B as its third argument. From $\mathrm{Tr}\,(\rho^B)^2 = 1/2$ we can easily see that the subsystem B is in a *mixed* state. Due to the symmetry of the Bell state, the same is true also for the subsystem A. Therefore, although the original composite system was in a completely determined (pure) state, the two subsystems A and B by themselves are statistical mixtures of states.

2.2. Schmidt Decomposition of a Pure State. In our second example, we demonstrate the so-called Schmidt decomposition of a pure state. It is well known that any pure bipartite n-qubit system AB which is described by a state vector $|\Psi^{AB}\rangle$ can be written in the form

$$|\Psi^{AB}\rangle = \sum_i \sqrt{p_i}\,|\psi_i^A\rangle\,|\phi_i^B\rangle, \tag{1}$$

where $\{|\psi_i^A\rangle\}$ and $\{|\phi_i^B\rangle\}$ denote orthonormal bases of the subsystems A and B and $\sqrt{p_i}$ is the real, non-negative Schmidt coefficient fulfilling $\sum_i p_i = 1$. This is widely known as the *Schmidt decomposition* [6]. Here, we would like to use the Feynman package to demonstrate such a decomposition for an example state where the result is not too obvious any more and which would require some straightforward but tedious manual calculations:

$$|\Psi^{AB}\rangle = \frac{1+\sqrt{6}}{2\sqrt{6}}\,|00\rangle + \frac{1-\sqrt{6}}{2\sqrt{6}}\,|01\rangle + \frac{\sqrt{2}-\sqrt{3}}{2\sqrt{6}}\,|10\rangle + \frac{\sqrt{2}+\sqrt{3}}{2\sqrt{6}}\,|11\rangle. \tag{2}$$



Bloch sphere visualization of qubit B in Subsec. 2.2, as returned by the `Maple` command line `Feynman_plot_Bloch_vector(Psi_AB, 1);`

Similar to our previous example, first, we have to define a `qregister` structure which describes the state $\left|\Psi^{AB}\right\rangle$. In `Feynman`, this is done by typing in a linear combination of computational basis states (`cbs`):

```
> Psi_AB := Feynman_set_qregister((1 + sqrt(6))/(2*sqrt(6))*cbs("00") + (1 -
  sqrt(6))/(2*sqrt(6))*cbs("01") + (sqrt(2) - sqrt(3))/(2*sqrt(6))*cbs("10") +
  (sqrt(2) + sqrt(3))/(2*sqrt(6))*cbs("11")):
```

Note that in order to save space the output of the previous command has been suppressed by using «`:`» (instead of «`;`») at the end of the line. Now the actual decomposition is done by the `Feynman_decompose` command:

```
> Feynman_decompose("Schmidt", Psi_AB);

                    [ 1/2  1/2]  [  1/2 ]           [ 1/2]  [ 1/2]
                    [2    3   ]  [ 2    ]           [3   ]  [2   ]
              1/2   [---------]  [ ---- ]           [----]  [----]
             3      [    3    ]  [  2   ]           [ 3  ]  [ 2  ]
           [[----, [[         ], [      ]]], [1/2, [[    ], [    ]]]]
              2     [    1/2  ]  [   1/2]           [ 1/2]  [ 1/2]
                    [   3     ]  [  2   ]           [6   ]  [2   ]
                    [ - ----  ]  [- ----]           [----]  [----]
                    [    3    ]  [   2  ]           [ 3  ]  [ 2  ]
```

Here, the output is given in the form $\left[\left[\sqrt{p_1}, \left[\left|\psi_1^A\right\rangle, \left|\phi_1^B\right\rangle\right]\right], \left[\sqrt{p_2}, \left[\left|\psi_2^A\right\rangle, \left|\phi_2^B\right\rangle\right]\right]\right]$ so that each list element represents one term of the sum in Eq. (1). From the fact that there is more than just one term in the sum (i.e. the *Schmidt rank* is greater than one) we can see that the original state $\left|\Psi^{AB}\right\rangle$ was entangled (but not maximally entangled). Similar to the first example in Subsec. 2.1 with the (maximally entangled) Bell state, the entanglement between A and B could have been also revealed by tracing over either of the two qubits A or B and studying the remaining qubit. For instance, in `Maple`'s graphic mode, the state of the second qubit B can be visualized conveniently in the well-known Bloch sphere, by using the command Feynman_plot_Bloch_vector() which internally traces out all remaining qubits which are not shown.

```
> Feynman_plot_Bloch_vector(Psi_AB, 1);
```

The resulting output graphic for the second qubit is shown in the figure. From this figure, it can be seen that the Bloch vector of the reduced density operator ρ^B has a length less than 1 which indicates that system B is in a mixed (but not maximally mixed) state, as should be expected for a subsystem of an entangled state.

REFERENCES

1. *Shor P. W.* Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer // SIAM J. Comp. 1997. V. 26, No. 5. P. 1484–1509.

2. *Wallace J.* Quantum Computer Simulators — A Review // Intern. J. Comp. Anticipatory Syst. (CASYS). 2000. V. 10. P. 230–245.

3. *De Raedt H., Michielsen K.* Computational Methods for Simulating Quantum Computers. quant-ph/0406210.

4. *Blum K.* Density Matrix Theory and Applications. 2nd ed. N. Y.: Plenum Press, 1996.

5. *Breuer H.-P., Petruccione F.* The Theory of Open Quantum Systems. Oxford Univ. Press, 2002.

6. *Nielsen M. A., Chuang I. L.* Quantum Computation and Quantum Information. Cambridge: Cambridge Univ. Press, 2000.

7. *Radtke T., Fritzsche S.* Simulation of n-Qubit Quantum Systems — I. Quantum Registers and Quantum Gates // Comp. Phys. Commun. 2005 (submitted).