International Journal of Theoretical Physics, Vol. 43, No. 6, June 2004 (© 2004)

Quantum Computation of Fuzzy Numbers

Bart D'Hooghe,[1,4] Jarosław Pykacz,[2] and Roman R. Zapatrin[3]

Received

We study the possibility of performing fuzzy set operations on a quantum computer. After giving a brief overview of the necessary quantum computational and fuzzy set theoretical concepts we demonstrate how to encode the membership function of a digitized fuzzy number in the state space of a quantum register by using a suitable superposition of tensor product states that form a computational basis. We show that a standard quantum adder is capable to perform Kaufmann's addition of fuzzy numbers in the course of only one run by acting at once on all states in the superposition, which leads to a considerable gain in the number of required operations with respect to performing such addition on a classical computer.

KEY WORDS: quantum computation; fuzzy numbers; Kaufmann addition.

1. INTRODUCTION

After the discovery of Shor's factorization algorithm for quantum computers (e.g., Ekert and Jozsa, 1996; Shor, 1994) many scientists searched for other quantum algorithms which would allow to perform calculations on a quantum computer more quickly than on a classical digital computer. This paper studies the possibility of performing fuzzy set operations on a quantum computer. The idea originated from the observation that fuzzy set theory and quantum theory are related in the sense that quantum probabilities of experimental outcomes show a fuzzy character (Pykacz, 1998). This quantum fuzziness was demonstrated by showing that one can represent any quantum logic with an ordering set of states by a suitably chosen family of fuzzy sets using Giles (Łukasiewicz) fuzzy set connectives to represent meet and join operations (Pykacz, 1987, 1994, 2000a,b). In this paper we will show that one can perform Kaufmann's addition of fuzzy numbers (Kaufmann, 1975) on a quantum computer in a much more efficient way with respect to performing such addition on a classical computer.

[1] FUND-CLEA, Vrije Universiteit Brussel, Belgium.
[2] Instytut Matematyki, Uniwersytet Gdański, Poland.
[3] Quantum Information Group, Fondazione ISI, Torino, Italy.
[4] To whom correspondence should be addressed at FUND-CLEA, Vrije Universiteit Brussel, Belgium; e-mail: bdhooghe@vub.ac.be.

In Section 2 we recall the main concepts and ideas of quantum computation treated as a physical process and give examples of quantum gates used to construct a quantum adder. Section 3 is devoted to the standard fuzzy arithmetics and Kaufmann's addition of fuzzy numbers and shows how this is done on a classical computer. In the next section we show how to encode membership functions of fuzzy sets in the form of suitable superposition states of a quantum register in a number of steps that is linear in the size of the input. We also demonstrate that Kaufmann's addition of fuzzy sets may be performed by a standard quantum adder already described in the literature. The famous quantum parallelism allows to perform all calculations involved in computing the Kaufmann's sum of two digitized fuzzy numbers in the course of only one step reducing considerably the number of operations to be performed. Although in principle one has to perform many measurements on the output state of the quantum register in order to obtain the membership function of the result, in some cases the demand of a finite precision of the result allows to obtain a considerable computational gain. It is an open question whether other fuzzy set operations can be calculated efficiently on a quantum computer and whether more sophisticated methods for obtaining (useful characteristics of) the output membership function can be developed in the future.

2. WHAT IS A QUANTUM COMPUTER?

2.1. Computation is a Physical Process

Computation can be described mathematically using the concept of an abstract Turing machine, but actual computation is always performed by some real physical system. The input is encoded by a suitably chosen initial state of a computer's register, after which a processor induces changes of the state of the register till the final state of the register is obtained and the algorithm stops. An output of the computational process is obtained by decoding the final state of the register. Using this view one can compare computational abilities of classical and quantum computers that follow from their very physical nature.

Classical computers are characterized by a register that is a set of bistable classical physical systems (switches, wires, magnets, etc.) able to store binary units (bits) of information: "0" or "1." The processor forces a transition of the initial state of the register to the final state according to the laws of classical physics. Of course, in order to understand really the working of modern classical computers one has to use quantum description for semiconductors used in the computer, but still the states of the register used to store information are classical. Therefore, a classical l-bit register can be in only one of its possible 2^l states at a time and a classical processor forces its transition to another such state, which is described mathematically as the action of Boolean functions on bit strings. Quantum computers on the other hand have a register which is a set of bistable quantum physical systems (two-level atoms, spin-1/2 particles, photons in orthogonal polarization states, etc.) able to store quantum binary units (qubits) of information. The state of each qubit is represented by a vector in a two-dimensional complex Hilbert space:

$$|q\rangle = c_0|0\rangle + c_1|1\rangle,\ c_0, c_1 \in \mathbb{C},\ |c_0|^2 + |c_1|^2 = 1 \tag{1}$$

in which $|q\rangle$ represents neither a statistical mixture of "falsity" and "truth" nor some intermediate truth value between "0" and "1". The processor forces transition of the initial state of the register to the final state according to the laws of quantum mechanics, which is described mathematically as the action of unitary transformations on the qubit strings representing the state of the quantum register. As such, a quantum l-qubit register can be not only in any of the 2^l "classical" states: $|0\rangle = |00\ldots0\rangle, |1\rangle = |00\ldots1\rangle, \ldots, |2^l - 1\rangle = |11\ldots1\rangle$, but also in any superposition state $|s\rangle$:

$$|s\rangle = \sum_{i=0}^{2^l-1} c_i|i\rangle,\ c_0, \ldots c_{2^l-1} \in \mathbb{C},\ \sum_{i=0}^{2^l-1} |c_i|^2 = 1, \tag{2}$$

and a quantum processor in a single run performs a unitary transformation simultaneously on each qubit string that is in this superposition, thus acting as a huge family of parallel processors, which is the essence of the famous "quantum parallelism."

2.2. Virtues and Drawbacks of Quantum Computers

First of all, it has been shown that quantum computers are universal, i.e., quantum computers, in principle, can do everything that classical computers can do (and, hopefully, much more . . .). Also, quantum parallelism can enormously speed up computations, e.g., Shor's factorization algorithm implemented on a quantum computer should allow to factorize 100-digit numbers in several seconds, whereas the most powerful of existing classical computers could not accomplish this task in a time shorter than the lifetime of the Universe from the Big Bang till the present moment, because the best algorithms for classical computers for the factorization problem are exponential with respect to the size of the input.

On the other hand, quantum computing also has its drawbacks: In most cases there is no direct access to the final state of the register because any measurement of this state selects from the superposition state randomly only one specific value $i \in \{0, \ldots, 2^l - 1\}$ with probability $|c_i|^2$. Therefore, special techniques have to be developed in order to get useful information out of the quantum register. Moreover, the actual building of quantum computers poses great technological problems caused by decoherence due to interaction with the environment. These problems are unavoidable when dealing with ultrasensitive single quantum objects,

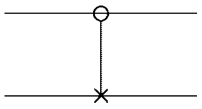
Fig. 1. Graphical representation of the controlled-NOT gate.

and although error correction techniques have been put forward, whether useful quantum computers can ever be actually built still remains an open question.

2.3. Elementary Quantum Logic Gates

Let us recall some examples of the most fundamental elementary quantum gates (see, e.g., Vedral *et al.*, 1996; Vedral and Plenio, 1998). The first is the controlled-NOT gate which acts on two qubits and negates the second bit iff the first one is "1". This gate induces, therefore, a unitary state transformation given by:

$$\begin{aligned} C_{\text{not}} : |00\rangle &\rightarrow |00\rangle \\ |01\rangle &\rightarrow |01\rangle \\ |10\rangle &\rightarrow |11\rangle \\ |11\rangle &\rightarrow |10\rangle \end{aligned}$$

This gate is usually represented by Fig. 1, in which the upper qubit is the control bit.

A modified version of this gate acts on three qubits negating the third one iff the first and the second are "1". It is called the Toffoli gate or the controlled–controlled-NOT gate (Fig. 2).

2.4. The Full Quantum Adder

Using these elementary quantum gates one can build a 1-bit quantum adder (Fig. 3).

By arranging a number of 1-bit quantum adders in a cascade (cf. Vedral *et al.*, 1996), one obtains the full quantum adder whose action can be represented by the unitary operation U:

$$|i\rangle \otimes |j\rangle \otimes |0\rangle \xrightarrow{U} |i\rangle \otimes |j\rangle \otimes |i+j\rangle.$$

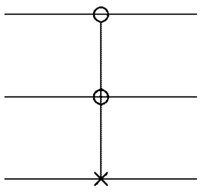
Fig. 2. Graphical representation of the Toffoli gate.

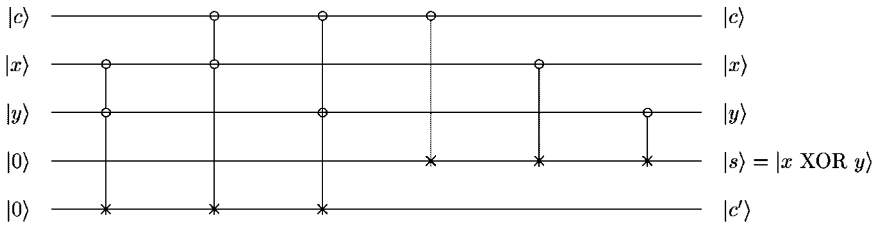

Fig. 3. The 1-bit quantum adder.

Let us note that the input is preserved in the output, which makes these operations reversible, as required by the quantum theory.

3. THE STANDARD FUZZY ARITHMETICS

3.1. Fuzzy Numbers

A fuzzy number is a convex and normalized fuzzy subset $\tilde{x}$ of the real line, i.e., such fuzzy subset of $\mathbb{R}$ that the convexity condition:

$$\forall_{u,v\in\mathbb{R}} \quad \forall_{\lambda\in[0,1]} \quad \tilde{x}(\lambda u + (1-\lambda)v) \geq \min(\tilde{x}(u), \tilde{x}(v)) \tag{3}$$

and the (fuzzy) normalization condition hold:

$$\exists! x \in \mathbb{R} \quad \text{such that} \quad \tilde{x}(x) = 1. \tag{4}$$

Let us note that this normalization condition is not a very stringent one because any membership function that attains its maximum can be (fuzzy) normalized by dividing it by its maximal value.

The standard way of defining arithmetic operations on fuzzy numbers utilizes the sup-min extension principle. For example, addition of two fuzzy numbers is usually defined as follows:

$$(\tilde{x} + \tilde{y})(z) = \sup_{u+v=z} \min(\tilde{x}(u), \tilde{y}(v)). \tag{5}$$

The sup-min operations on fuzzy numbers, although the most popular, are not the only one considered in the literature (see, e.g., Dubois and Prade, 1980). Another way of adding fuzzy numbers was considered by Kaufmann in his book (Kaufmann, 1975) already in 1975. He used a probabilistic method of extending addition to fuzzy numbers by means of an ordinary convolution:

$$(\tilde{x} + \tilde{y})(z) = \int_0^z \tilde{x}(u)\tilde{y}(z-u)du \tag{6}$$

3.2. Addition of Fuzzy Numbers on a Classical Computer

The most general way in which operations on fuzzy sets can be emulated on a conventional digital computer consists in digitizing all membership functions of the input and performing the required operations pointwisely in order to get the digitized membership function of the output.

In such case we deal with numbers of the form: $\tilde{a} = \{\tilde{a}(0), \tilde{a}(1), \ldots, \tilde{a}(n)\}$, where we admit that some of the values $\tilde{a}(i)$ may equal zero. In this case Kaufmann's addition (6) takes the discrete form:

$$(\tilde{a} + \tilde{b})(k) = \sum_{i+j=k} \tilde{a}(i)\tilde{b}(j). \tag{7}$$

It can be easily checked that the total number of arithmetic operations that a conventional digital computer has to perform in order to obtain the membership function of the sum of two fuzzy numbers whose supports consist of n points is of the order n^2.

4. QUANTUM FUZZY ARITHMETICS

4.1. Encoding the Input

In this subsection we shall show how to represent digitized fuzzy numbers by suitable superpositions (2) of states of the register that form its computational basis.

The state $|s\rangle = \sum_{i=0}^{2^l-1} c_i |i\rangle$ of the formula (2) can be thought of as representing a digitized fuzzy number $\widetilde{s}$ with a support consisting of 2^l integers $\{0, 1, \ldots, 2^l - 1\}$ and a membership function $\widetilde{s}(i) = |c_i|^2$ for any $i \in \{0, 1, \ldots, 2^l - 1\}$. Of course such membership function is normalized in probabilistic, not fuzzy sense: $\sum_{i=0}^{2^l-1} |c_i|^2 = 1$, while in general $\max_i |c_i|^2 < 1$, but, according to our remark about the fuzzy normalization condition made in subsection 3.1, this should yield no problems.

Suppose now that we want to represent in this way a specific digitized fuzzy number with predefined values $\tilde{a}(i)$ of its membership function. This can be done by applying the general algorithm described by Zalka (1998). In our case this algorithm is particularly simple and we shall explain it on the following example.

Example. Let us represent in the form (2) a fuzzy number whose support consists of four integers: 0, 1, 2, 3, and whose membership function takes values: $\tilde{a}(0) = 0.1$, $\tilde{a}(1) = 0.3$, $\tilde{a}(2) = 0.5$, $\tilde{a}(3) = 0.1$. As we see, this membership function is already normalized probabilistically: $\sum_{i=0}^{3} \tilde{a}(i) = 1$. If it was not, one should normalize it probabilistically dividing it by $\sum_{i=0}^{3} \tilde{a}(i)$ before applying Zalka's algorithm.

The algorithm goes as follows: Since the support of our number consists of four points, the superposition (2) we aim at should contain four terms and might be of the form

$$\pm\sqrt{0.1}|00\rangle \pm \sqrt{0.3}|01\rangle \pm \sqrt{0.5}|10\rangle \pm \sqrt{0.1}|11\rangle, \tag{8}$$

since $(\pm\sqrt{0.1})^2 = 0.1 = \tilde{a}(0)$, etc. In order to get such superposition from the state $|00\rangle$ we divide the membership function of our number into two halves, calculate the sums of its values in the left half getting 0.4, and in the right half getting 0.6, and apply the rotation by arctan $\frac{\sqrt{0.6}}{\sqrt{0.4}}$ to the first qubit in the tensor product $|00\rangle = |0\rangle \otimes |0\rangle$:

$$\left[\begin{pmatrix} \sqrt{0.4} & \sqrt{0.6} \\ -\sqrt{0.6} & \sqrt{0.4} \end{pmatrix} \otimes I\right](|0\rangle \otimes |0\rangle) = \sqrt{0.4}(|0\rangle \otimes |0\rangle) - \sqrt{0.6}(|1\rangle \otimes |0\rangle) \tag{9}$$

In the next (and, in our case, the last) step we apply to the result of the previous step unitary transformation $|0\rangle\langle 0| \otimes U_0 + |1\rangle\langle 1| \otimes U_1$, where U_0 is the rotation by arctan $\frac{\sqrt{0.3}}{\sqrt{0.1}}$, and U_1 is the rotation by arctan $\frac{\sqrt{0.1}}{\sqrt{0.5}}$:

$$\begin{aligned} &(|0\rangle\langle 0| \otimes U_0 + |1\rangle\langle 1| \otimes U_1)[\sqrt{0.4}(|0\rangle \otimes |0\rangle) - \sqrt{0.6}(|1\rangle \otimes |0\rangle)] \\ &\quad = \sqrt{0.4}(|0\rangle \otimes U_0|0\rangle) - \sqrt{0.6}(|1\rangle \otimes U_1|0\rangle). \end{aligned} \tag{10}$$

Taking into account that

$$U_0|0\rangle = \frac{\sqrt{0.1}}{\sqrt{0.4}}|0\rangle - \frac{\sqrt{0.3}}{\sqrt{0.4}}|1\rangle \tag{11}$$

and

$$U_1|0\rangle = \frac{\sqrt{0.5}}{\sqrt{0.6}}|0\rangle - \frac{\sqrt{0.1}}{\sqrt{0.6}}|1\rangle \tag{12}$$

we get from (10)

$$\begin{aligned} &|0\rangle \otimes (\sqrt{0.1}|0\rangle - \sqrt{0.3}|1\rangle) - |1\rangle \otimes (\sqrt{0.5}|0\rangle - \sqrt{0.1}|1\rangle) \\ &\quad = \sqrt{0.1}|00\rangle - \sqrt{0.3}|01\rangle - \sqrt{0.5}|10\rangle + \sqrt{0.1}|11\rangle \end{aligned} \tag{13}$$

which is the desired result. Of course if this step was not the last one, the coefficients in the rotation matrices should be obtained by calculating, respectively, the sums of values of the membership function in the "left half" and the "right half" of each of the "halves" from the previous step, and proceeding with such rotations until the desired result is obtained. It can be easily checked that if the support of a fuzzy number to be encoded consists of $n = 2^l$ integers, i.e., this number can be encoded by using qubit strings of the length l, the required number of additions necessary to calculate rotation coefficients equals $2^l - 2 = n - 2$. Let us note that this number is linear with respect to n while the total number of arithmetic operations performed by a classical digital computer in order to calculate Kaufmann's sum of two fuzzy numbers whose support consists of n points is quadratic with respect to n.

4.2. Action of the Quantum Processor

Suppose now that we have to add two fuzzy numbers $\tilde{a} = \tilde{a}(i)$ and $\tilde{b} = \tilde{b}(j)$ that are represented by $|a\rangle = \sum_i \varphi(i)|i\rangle$, $|b\rangle = \sum_j \psi(j)|j\rangle$, where $|\varphi(i)|^2 = \tilde{a}(i)$ and $|\psi(j)|^2 = \tilde{b}(j)$. We feed the following input to the adder

$$|a\rangle \otimes |b\rangle \otimes |0\rangle = \sum_{i,j} \varphi(i)\psi(j)|i\rangle \otimes |j\rangle \otimes |0\rangle. \tag{14}$$

The quantum adder acts on the states of the computational basis as follows: $|i\rangle \otimes |j\rangle \otimes |0\rangle \xrightarrow{U} |i\rangle \otimes |j\rangle \otimes |i + j\rangle$. By substituting this in (14) and rearranging the sum with respect to the last factor of the tensor product we obtain:

$$\sum_{i,j} \varphi(i)\psi(j)|i\rangle \otimes |j\rangle \otimes |i + j\rangle = \sum_k \sum_{i+j=k} \varphi(i)\psi(j)|i\rangle \otimes |j\rangle \otimes |k\rangle. \tag{15}$$

The probabilities of getting the particular values of the result are obtained by representing the state (15) as a projector in the tensor product space $H \otimes H \otimes H$ and taking the partial trace over the two first spaces in the tensor product:

$$\sum_{k,k'} \sum_{i+j=k} \sum_{i'+j'=k'} \delta_{ii'}\delta_{jj'}\varphi(i)\overline{\varphi(i)}\psi(j)\overline{\psi(j)}|i\rangle\langle i'| \otimes |j\rangle\langle j'| \otimes |k\rangle\langle k'|, \tag{16}$$

which yields the following vector:

$$\sum_k \sum_{i+j=k} |\varphi(i)|^2|\psi(j)|^2|k\rangle. \tag{17}$$

The formula (17) interpreted directly in terms of values of the membership function shows that the addition of two digitized fuzzy numbers realized by the standard quantum adder is exactly Kaufmann's addition (7).

4.3. Getting the Output

Because we have no direct access to the output state of the quantum register, if the task is to reconstruct the entire membership function of the result, we would have to repeat the addition many times, loosing in this way a big part of the gain that follows from changing the classical adder to the quantum one. However, because most of computations performed with fuzzy data finishes with some defuzzification procedure, we may assume that the task is to find the barycenter of the membership function of the output. The x-coordinate of this barycenter is just the mean value of the probability distribution of getting particular results of measurements performed on the quantum register that obtained its final state after consecutive runs of the quantum adder. Because the standard error of the mean value obtained in the course of k runs is proportional to $1/\sqrt{k}$, we see that if the desired accuracy of obtaining the "defuzzyfied result" is not too high, the number k may be much smaller than n^2, i.e., the number of arithmetic operations that a classical computer has to perform in order to get the membership function of the output (which still would have to be augmented if the final task were to find the barycenter). For example, if the assumed accuracy of finding the barycenter of the output is 10%, and the support of two digitized fuzzy numbers to be added consists of $n = 100$ points, then $k \simeq 100$, which is much smaller than the number of operations ($n^2 = 10.000$) that a classical computer should perform in order to get the same result.

5. CONCLUSIONS

Due to the famous quantum parallelism quantum computers may evaluate, in the course of a single run, whole membership functions of fuzzy sets instead of evaluating them "point by point." This, at least in some specific situations, would allow to considerably diminish the number of computational steps required to get the output in comparison to what could be achieved while using classical computers. For the moment it remains an open question whether also other fuzzy set operations, e.g., the most frequently used sup-min operations, could be calculated on a quantum computer more efficiently than on a classical computer, and whether better techniques can be developed to obtain useful information without having to measure the whole membership function, in a way that preserves the computational speedup of the quantum computer.

ACKNOWLEDGMENTS

The main part of the paper was written when all the three authors met in the Center Leo Apostel (CLEA), Brussels in September 2000. The authors are deeply indebted to CLEA for providing them with the facilities that stimulated their work. BD'H and JP were supported by the grant under the Polish–Flemish Research Project 127/E-335/S/2000 and RRZ by the Starlab, Brussels, Q-ACTA Project and the research grant "Universities of Russia." B. D'Hooghe is currently a Postdoctoral Fellow of the Fund for Scientific Research—Flanders (Belgium) (F.W.O.—Vlaanderen).

REFERENCES

Dubois, D. and Prade, H. (1980). *Fuzzy Sets and Systems. Theory and Applications*, Academic Press, New York.

Ekert, A. and Jozsa, R. (1996). Quantum computation and Shor's factoring algorithm. *Reviews of Modern Physics* **68**, 733–753.

Kaufmann, A. (1975). *Introduction to the Theory of Fuzzy Subsets, Vol. I*, Academic Press, New York.

Pykacz, J. (1987). Quantum logics as families of fuzzy subsets of the set of physical states. In *Preprints of the Second IFSA Congress, Tokyo, Vol. 2*, International Fuzzy Systems Association, Japan Chapter, Tokyo, 1987, pp. 437–440.

Pykacz, J. (1994). Fuzzy quantum logics and infinite-valued Łukasiewicz logic. *International Journal of Theoretical Physics* **33**, 1403–1416.

Pykacz, J. (1998). Fuzzy quantum logics as a basis for quantum probability theory, *International Journal of Theoretical Physics* **37**, 281–290.

Pykacz, J. (2000a). Łukasiewicz operations in fuzzy set and many-valued representation of quantum logics. *Foundations of Physics* **37**, 1503–1524.

Pykacz, J. (2000b). Quantum logic as a basis for computations. *International Journal of Theoretical Physics* **39**, 839–850.

Shor, P. W. (1994). Algorithms for quantum computation: Discrete logarithms and factoring. In *Proceedings of the 35th Annual Symposium on the Foundations of Computer Science*, S. Goldwasser, ed., IEEE Computer Society Press, Los Alamitos, CA, pp. 124–134.

Vedral, V., Barenco, A., and Ekert, A. (1996). Quantum networks for elementary arithmetic operations, *Physical Review A* **54**, 147–153.

Vedral, V. and Plenio, M. B. (1998). Basics of quantum computation. *Progress of Quantum Electronics* **22**, 1–40; see also e-print: quant-ph/9802065.

Zalka, C. (1998). Simulating quantum systems on a quantum computer. *Proceedings of the Royal Society of London Series A* **454,** 313–32; see also e-print: quant-ph/9603026.