T I T L E P A G E

QUANTUM COMPUTATION AND QUANTUM COMMUNICATION: Theory and Experiments

Mladen Pavičić (Mladen Pavicic)

University of Zagreb, Zagreb, Croatia

Springer

C O P Y R I G H T P A G E

Mladen Pavicic
University of Zagreb, Croatia
Consulting Editor: D. R. Vij
Library of Congress Cataloging-in-Publication Data
Pavicic, Mladen. Quantum computation and quantum communication : theory and experiments
Mladen Pavicic.
p. cm.
Includes bibliographical references.
ISBN 10 0-387-24412-3
ISBN 13 978-0387-24412-9
ISBN 0-387-28900-2 (e-book) 1. Quantum computers 2. Quantum theory. I. Title
QA76.889.P38 2005 2005051725

Printed in the United States of America
9 8 7 6 5 4 3 2 1 SPIN 11051145
springeronline.com

D E D I C A T I O N

Dedicated to the reader.

C O N T E N T S

P r e f a c e

The attraction of quantum computation and quantum communication theory and experiments lies in the fact that we engineer both them themselves and the quantum systems they treat. This approach has turned out to be very resilient. Driven by the final goal of calculating exponentially faster and communicating infinitely more securely than we do today, as soon as we encounter a limitation in either a theory or experiment, a new idea around the no-go emerges. As soon as the decoherence "demon" threatened the first computation models, quantum error correction theory was formulated and applied not only to computation theory but also to communication theory to make it unconditionally secure. As soon as liquid-state nuclear magnetic resonance experiments started to approach their limits, solid-based nuclear spin experiments - the Kane computer - came in. As soon as it was proved that it is theoretically impossible to completely distinguish photon Bell states, three new approaches appeared: hyperentanglement, the use of continuous variables, and the Knill-Laflamme-Milburn proposal. There are many more such examples.

What facilitated all these breakthroughs is the fact that at the present stage of development of quantum computation and communication, we deal with elementary quantum systems consisting of several two-level systems. The complexity of handling and controlling such simple systems in a laboratory has turned out to be tremendous, but the basic physical models we follow and calculate for the systems themselves are not equally intricate. We could say that the theory of the field leads the experiments in a particular way-with each new model we put forward and apply in the laboratory, we also build up and widen the theory itself. Therefore, we cannot just proceed with assembling quantum computers and quantum networks. We also have to use mathematical models to understand the physics of each step on the road to our goal.

As a consequence, both mathematics and physics are equally essential for any approach in the field and therefore for this book as well. The mathematics used in the book is a tool, but an indispensable tool because the physics of quantum computation and communication theory and their experiments cannot be grasped without good mathematical models. When we describe an experiment many times, we may get used to it, but this does not mean we are more at home with the principles and models behind it. This is

why I have chosen to make this book an interplay between mathematics and physics. The idea of the book is to present those details that are used the most often both in theory and experiment and to dispense with many inessential ones. Also, the book is not conceived as a textbook, at least not as a primary one, but more as a guide to a better understanding of theory and experiments by coming back to the same concepts in different models and elaborations. Clear physical ideas make any formalism easy.
Mladen Pavicic

Introduction

Two predictions are cited particularly often whenever one talks or writes about the history or future of computing. One of these is more and more wrong, and the other is less and less right, and they both teach us how to use theoretical opportunities to find new technologies.

The first prediction, a beloved opening of speeches and papers, was made by the head of the electromagnetic relay calculator at Harvard, Howard Aiken, in 1956: "If it should turn out that the basic logics of a machine designed for the numerical solution of differential equations coincide with the logics of a machine intended to make bills for a department store, I would regard this as the most amazing coincidence that I have ever encountered" [Anonymous, 1997]

The amazing "coincidence" did happen and happens more and more every day, tempting us to consider it a part of the history of computers that took its own unexpected course ("Only six electronic digital computers would be required to satisfy the computing needs of the entire United States," Howard Aiken said in 1947): a program and a machine, software and hardware, were interwoven at the beginning and then became more and more separated. At least it seems so when we look at the development of computer designs since Charles Babbage’s 1840s Analytical Engine. A program on punched cards or tapes and a machine for which the specific cards were made look inseparable, in contrast to today’s programs which we move throughout the World Wide Web and compile and execute on virtually any computer.

Yet Alan Mathison Turing (and also Alonzo Church, Stephen Cole Kleene, and Emil Post independently at the same time) had already proved in 1936 that the only possible course the history could have taken was the one it in fact took. Turing used what we now also cite often and call a *Turing machine* to prove that only the simplest calculus, such as a propositional algebra with a Boolean evaluation (true, false) and its main model a 0-1 Boolean algebra, is computable, i.e., effectively calculable [Turing 1936; Turing, 1937]. He (and others) also proved that real numbers are not computable, that there exists no algorithm with the help of which we can decide for every arithmetical sentence in finitely many steps whether it is true or false, etc. In other words, from the very start we only had Boolean algebra at our disposal, and once hardware was developed that could handle classical logic operations - such implementations of logic operations are called *logic gates* - the universal classical computer was born. The "only" thing one had to develop were "digital" algorithms and programs for all possible applications, i.e., the software for a universal computer. Everything - solving nonlinear differential equations, 3D modeling, speech recognition, and "making bills for a department store" - had to be reduced to a Boolean language. Since such a reduction imposes ever-growing speed and memory requirements upon the hardware, until mid-2002 we were witnessed quite the opposite situation than half a century ago: the software lagged behind the hardware, following the Wirth’s law: "Software gets slower faster than hardware gets faster."

Will this computing history repeat itself with quantum computers? Will quantum hardware start to advance faster than quantum software (quantum algorithms) in the near future? In this book we shall try to learn how close we are to answering these questions.

The second prediction is known as *Moore’s Law*, or better yet, Moore’s laws, since there are many versions and varieties of the several formulations made by Gordon Moore of the Intel Corporation. One widespread rendering of the law, "The number of transistors on a single integrated-circuit chip doubles every 18 months" [Birnbaum and Williams, 2000], does not correspond to the historical data which show 26 months [Brenner, 2001]. Moore himself commented. "I never said 18 months. I said one year [in 1965], and then two years [in 1975]. One of my Intel colleagues changed it from the complexity of the chips to the performance of computers and decided that not only did you get a benefit from the doubling every two years but we were able to increase the clock frequency, too, so computer performance was actually doubling every 18 months. I guess that's a corollary of Moore's Law. Moore's Law has been the name given to everything that changes exponentially in the industry... If Al Gore invented the Internet, I invented the exponential" [Yang, 2000]

And this "exponential" element is what is essential for our development and what quantum computers are about. Apparently everything underlying the development of technology and society grows exponentially: research, information, production and organization complexity, and above all, the costs of keeping pace. So only an exponential increase of our computational and processing power and an exponential decrease of computer cost per processed bit could support such a development. Therefore, Moore's law was been kept as a guideline in the computer industry in past three decades and it has supported a global development during this period.

Gates in today's computers are switched on and off by about 1000 electrons. In 2010, the exponential Moore's Law would require that only about 10 electrons do the job. Miniaturization cannot go much further than that. It is true that many other possible roads could still keep up the pace for a few more years: insulating layers can be reduced in their thickness from the present 25 atoms to 4 or 5 atoms (wires connecting transistors in a chip already occupy more than 25% of its space); computing power can be increased by designing processors so as to contain execution units that process multiple instructions within one cycle; processors can rely on parallel compiling technology and use innovative software; and finally, chips can eventually get bigger by using reversible gates to avoid overheating. Still, by 2020 or 2025 computing technology will hit the quantum barrier, and if we want to support the growth of our technology and science beyond that point in time, we need to find a substitute for exponentially rising classical computational power by then. Actually, the exponential increase of the clock speed of processors (CPUs) already became linear in 2002 (see Fig. 3.1, p. 135), and an extensive patching activity onto classical hardware and software is currently under way in order to compensate for this lack of an exponential increase in speed (see p. 136).

Now that both Wirth's and Moore's laws are coming to an end, we should draw a moral from them. Wirth's law taught us that classical hardware development has prompted ever new software, and Moore's law taught us that this hardware development has followed an exponential trend of speed, memory, and lately of number of processors (multiple cores, multiple processors, clusters). Such an approach to computation will apparently change completely in the quantum realm. Quantum hardware is exponential in itself, and if we eventually succeed in making functional scalable quantum computers, we will dispense with the need for a steadily growing quantum hardware development - to make a quantum computer faster means to scale it up linearly or polynomially. We will also dispense with writing ever new software for faster and faster hardware. Once developed, quantum software (quantum algorithms) will simply scale up as we scale - and therefore speed up - quantum hardware.

The "exponential" is built into quantum hardware from its very first *qu*antum *bit* or *qubit*. Qubits, physically supported by single atoms, electrons, or photons, can superpose and entangle themselves so as to support an arbitrary number of states per unit. Recently devised algorithms - quantum software - relying on the exponential feature of quantum hardware have explicitly demonstrated how one can reduce important problems that are assumed to be exponentially complex, to polynomially complex tasks for quantum computers. This has opened a vast new interdisciplinary field of quantum computation and communication theories, together called quantum information theory, which along with its experimental

verifications are already taught at many universities and have resulted in several very successful textbooks.
The target of these courses, seminars, and textbooks is to teach and familiarize students and scientists with this new field - in which new research projects will keep opening for decades to come - and to help integrate the theory and experiments of quantum computation and communication into a would-be quantum network implementation. The goal of the book in front of the reader is the same; however, it allows her or him to digest the field "by reading." That means that there will be no homework and no exercises. Instead, most of the required details are elaborated within the main body of the book, and a polynomial complexity of reading is intended, optimally in one run.
So, a few words about the reader. She or he is expected to be familiar with higher mathematics and the basics of physics - in particular, quantum physics. The reader could be any former student who graduated in the technical or natural sciences, although an undergraduate student might also find many if not all sections of the book digestible. Students as well as specialists in the field might also find the nutshell approach of the book helpful and stimulating.

Chapter 1

BITS AND QUBITS: THEORY AND ITS IMPLEMENTATION

In 1936 several authors showed, in effect, that if a function is effectively calculable, then it is Turing computable and, of course, vice versa [Church, 1936c; Turing, 1936; Turing, 1937; Church, 1936a; Church, 1936b; Kleene, 1936; Post, 1936]. Turing concluded: We do not need to have an infinity of different machines doing different jobs. A single one will suffice. The engineering problem of producing various machines for various jobs is replaced by the office work of "programming" the universal machine to do these jobs
This statement does not mean that Turing envisioned the "universal computer" we have today, although he was well acquainted with the project of breaking the cryptographic codes of German messages carried out on the Colossus (the British "computer" at Bletchley Park, which operated from 1943 until the 1950s). His *universal Turing machine* is a "universal computer" only in the sense that it keeps to the standard digital (classical, 0-1) implementation, i.e., to the *bi*nary digi*ts*, or *bits*, of today’s hardware.

1.1 The Turing Machine vs. a Computing Machine

The software used by any classical computer must be based on what a Turing machine can confirm to be calculable, recursive, and decidable. A historical problem with the development of computers was that there were few calculus categories of the latter kind. The only types of calculus that Turing machines can show to be calculable are the simplest algebras with the simplest evaluations, such as propositional calculus with Boolean (true-false) evaluation, or 0-1 Boolean algebra. It can be shown that even the simplest propositional calculus with a nonordered evaluation [Pavicic and Megill, 1999] or simplest arithmetic with natural numbers [Hermes, 1969] is not calculable simply because such types of algebra are neither recursive nor decidable nor calculable. Directly, a Turing machine can only be used to *prove* that no mathematics we know from primary school can be literally run on it.
Turing machines, or any equivalent mathematical algorithms, are essential in order to decide whether a chosen problem is calculable or not, but we do not use them to write down a new program for, say, 3D modeling or speech recognition. Still, since there are many references to the Turing machine in the literature on quantum computing, let us provide some details [Hermes, 1969]. In doing so, we bear in mind

that Turing machines and all related concepts are "concepts of pure mathematics. It is however very suggestive to choose a technico-physical terminology suggested by the mental image of a machine" [Hermes, 1969, p.31].

The Turing machine is neither today’s "universal" computing machine - generally called a computer - nor a generator of new algorithms for the latter machine. Instead, it is simply a mathematical procedure to check whether a chosen algebra and/or calculus can or cannot be implemented into a computer. To show this, we present some details of the procedure. The details often appear in the literature without being put into the context of a final outcome and so are just left hanging, giving the impression of being building blocks for a computer, or an algorithm to be carried out on one. On the other hand, the notion of the classical Turing machine is rather important for understanding the role that the quantum Turing machine has in the theory of quantum computation.

INDEX

ON THE AUTHOR

Pavicic, Mladen [From MARQUISE *Who's Who in Science and Engineering*, 4th Ed., 1998-1999 and *Who's Who in the World*, 17th, 2000] Physicist, educator; b. Zagreb, Croatia; PhD in Physics, 1986; main asst. Univ. Zagreb, 1982-89, asst. prof., 1990-95, assoc. prof., 1996-2000, full prof., 2001- ; head sci. project Ministry of Sci., Zagreb, 1991-96; head sci. project *Quantum Computation and Quantum Communication*, Ministry of Sci., Zagreb, 1996-2001, head sci. project *Quantum Information Theory*, Ministry of Sci., Educ., and Sport, Zagreb, 2001-. Author: *Solved Problems in Physics*, 1982, 2nd edit., 1984; articles to prof. journals: *Phys. Rev. Lett.; Phys. Rev. A, D; J. Opt. Soc. Am. B; Opt. Commun.; J. Phys. A; Phys. Lett. A; Helv. Phys. Acta; Forschr. Phys.; Found. Phys.; Int. J. Theor. Phys.;* etc.; Grantee Alexander von Humboldt Found., Germany, Univ. Cologne, Germany, 1988-90; Tech. Univ. Berlin, Germany, 1993; Humboldt Univ. Berlin, Germany, 1995; Grantee Fulbright Senior Scholar

Research/Lect., USA, Univ. Maryland Baltimore County, UMBC, Baltimore, USA, 1999-2000; French Ministry Sci., Univ. Reims, France, 1992; Austrian Ministry Sci., Atom-Inst. of Austr. Univ., Vienna, Austria, 1993, 94, 95, 97; Max-Planck Gesel., Germany, Humboldt Univ. Berlin, Germany, 1996; E. Schrodinger Inst., Vienna, Austria, 2000; Mem. Int. Quantum Structures Assn. (co-founder, nominating com. 1992-94), USA; Croatian Humboldt-Club (president, 2002-); Croatian Phys. Soc.; European Phys. Soc.; Am. Phys. Soc.; Opt. Soc. Am.; Achievements include proof of Pauli nonuniqueness for real states, discovery of polarization correlation between beams of unpolarized light, discovery of polarization entanglement of two unpolarized photons that nowhere interacted, discovery of a nondistributive model for classical logic, co-discovery of a nonorthomodular model for quantum logic, co-formulation of a resonator interaction-free detection, discovery of an interaction free destruction of atom interference pattern, and co-discovery of exhaustive algorithms for generating arbitrary Kochen-Specker vectors.

BACK COVER (EDITORIAL REVIEW)

QUANTUM COMPUTATION AND QUANTUM COMMUNICATION: Theory and Experiments

Mladen Pavicic

The field of quantum computing has experienced rapid development and many different experimental and theoretical groups have emerged worldwide.This book presents the key elements of quantum computation and communication theories and their implementation in an easy-to-read manner for readers coming from physics, mathematics and computer science backgrounds. Integrating both theoretical aspects and experimental verifications of developing quantum computers, the author explains why particular mathematical methods, physical models and realistic implementations might provide critical steps towards achieving the final goal - constructing quantum computers and quantum networks. The book serves as an excellent introduction for new researchers and also provides a useful review for specialists in the field.

Springer

springeronline.com

ISBN 0-387-24412-3

Misprints (Errata)

p. 22, the lower row of the matrix given by Eq. (1.14) should read:

$$0 \quad e^{i\delta_y}$$

p. 22, the line above Eq. (1.16) should read:

polarization at $-45°$

p. 22, the second line above Eq. (1.17) should read:

ization at $+45°$, and a left-hand circular polarization into a right-hand

p. 114, the line below Eq. (2.67) should read:

where $\omega_{eg} = (E_e - E_g)/\hbar$ and $R_{eg} = D_{eg}E_0/\hbar = R_{ge}$, where D_{eg}, a spatial

p. 114, 5th line from bottom should read:

integral of $\langle e|\boldsymbol{\mu}\cdot\mathbf{E}|g\rangle$, is called the *atomic dipole matrix element*—it is

p. 114, 2nd line from bottom should read:

only $c_g(t)$. This is because $D_{gg} = D_{ee} = 0$. Physically, it means that $|e\rangle$

p. 115, the line above Eq. (2.68) should read:

Eqs. (2.66) and (2.67) read [Demtröder, 1996, 2.6.6]

p. 136, 11th line from the top should read:

cal computing we must turn to software but at the same time we should be

p. 139, the first half of the 8th line from bottom should read:

stimulated emission of two photons ($p_{\rm i3}$,$p_{\rm o3}$ in Fig. 3.2) any more,

p. 147, Eq. (3.19) should read:

$$a|g_1\rangle_{\rm A} + b|g_2\rangle_{\rm A}, \tag{3.19}$$

p. 147, Eq. (3.20) should read:

$$a|g_1\rangle_{\rm B} + b|g_2\rangle_{\rm B}, \tag{3.20}$$

p. 147, the 5th line from bottom should read:

polarizations as given by Eq. (1.13). We cannot use linearly polarized

p. 148, the 1st line of the caption of Figure 3.6 should read:

Figure 3.6. Levels of ^{87}Rb ($|g\rangle$, $|e\rangle$), laser beams (Ω), atom–cavity couplings (g),

p. 148, the 9th line from bottom should read:

$|e_3\rangle_B$ (Ω_{B1} and Ω_{B2} because of their opposite detunings) by means of

p. 149, Eq. (3.25) should read:

$$|\Psi\rangle_B = |g_1, R\rangle_B + |g_2, L\rangle_B. \tag{3.25}$$

p. 151, Eq. (3.33) should read:

$$|\phi^+\rangle \to \frac{1}{\sqrt{2}}(|H_1\rangle|H_2\rangle + |V_1\rangle|V_2\rangle),\ \ |\phi^-\rangle \to \frac{1}{\sqrt{2}}(|H_1\rangle|V_2\rangle + |V_1\rangle|H_2\rangle). \tag{3.33}$$

p. 166, the 3rd line of the caption of Figure 3.16 should read:

enter the cavity, and we have $0 \to 0$ and $1 \to 1$. (b) The atom is in the $g2$ state and

p. 166, the 5th line of the caption of Figure 3.16 should read:

and we have $0 \to 1$ and $1 \to 0$. ABS are highly asymmetrical beam splitters with

p. 167, the last line of the 2nd paragraph should read:

that have verified them.

p. 197, Eq. (3.127) should read:

$$(\forall A, B \in \mathcal{L})(\exists m \in S)((m(A) = 1 \ \Rightarrow\ m(B) = 1) \ \Rightarrow\ A \le B)\,. \tag{3.127}$$