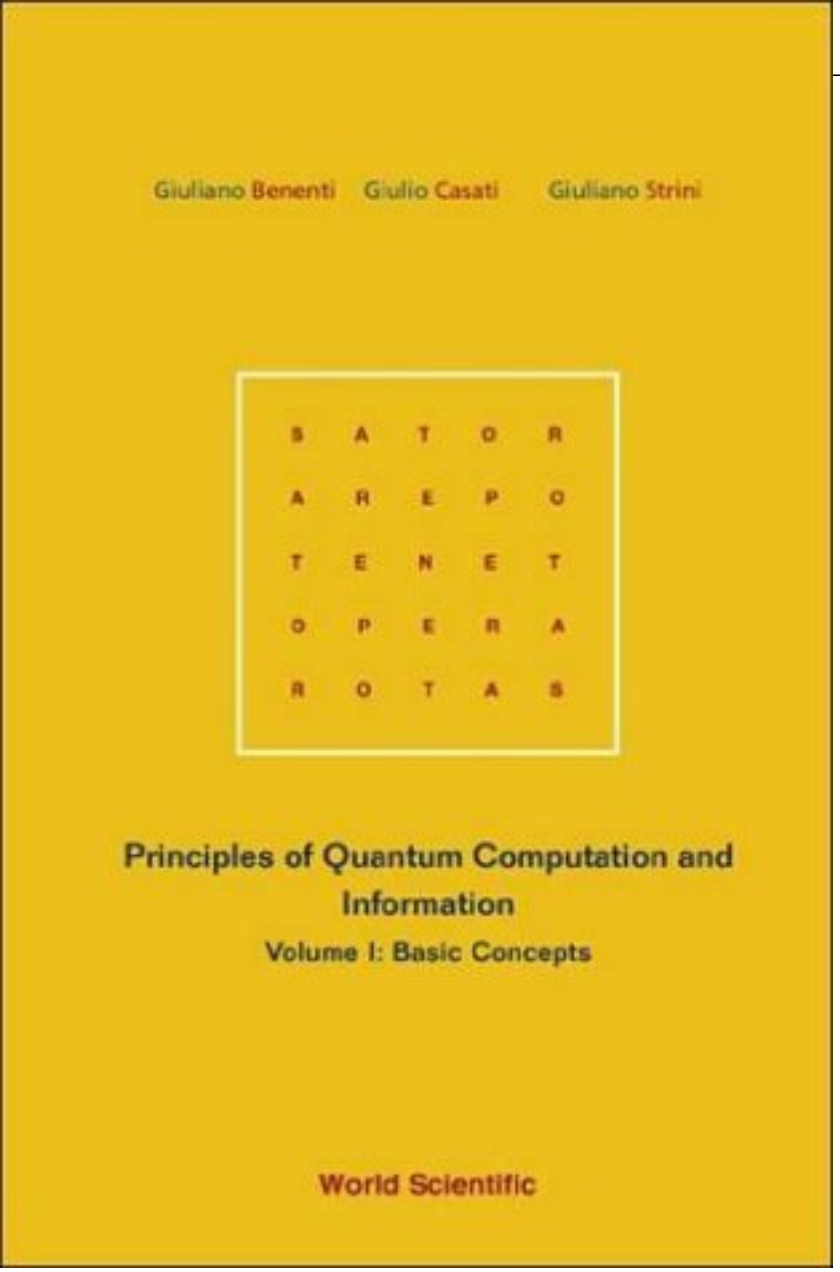
Giuliano Benenti Giulio Casati Giuliano Strini
S A T O R
A R E P O
T E N E T
O P E R A
R O T A S
Principles of Quantum Computation and Information
Volume I: Basic Concepts
World Scientific

To Silvia
g.b.

To my wife for her love and encouragement
g.c.

To my family and friends
g.s.

Preface

Purpose of the book

This book is addressed to undergraduate and graduate students in physics, mathematics and computer science. It is written at a level comprehensible to readers with the background of a student near to the end of an undergraduate course in one of the above three disciplines. Note that no prior knowledge either of quantum mechanics or of classical computation is required to follow this book. Indeed, the first two chapters are a simple introduction to classical computation and quantum mechanics. Our aim is that these chapters should provide the necessary background for an understanding of the subsequent chapters.

The book is divided into two volumes. In volume I, after providing the necessary background material in classical computation and quantum mechanics, we develop the basic principles and discuss the main results of quantum computation and information. Volume I would thus be suitable for a one-semester introductory course in quantum information and computation, for both undergraduate and graduate students. It is also our intention that volume I be useful as a general education for other readers who would like to learn the basic principles of quantum computation and information and who have the basic background in physics and mathematics acquired in undergraduate courses in physics, mathematics or computer science.

Volume II deals with various important aspects, both theoretical and experimental, of quantum computation and information. This volume necessarily contains parts that are more technical or specialized. For its understanding, a knowledge of the material discussed in the first volume is necessary.

General approach

Quantum computation and information is a new and rapidly developing field. It is therefore not easy to grasp the fundamental concepts and central results without having to face many technical details. Our purpose in this book is to provide the reader interested in this field with a useful and not overly heavy guide. Therefore, mathematical rigour is not our primary concern. Instead, we have tried to present a simple and systematic treatment, such that the reader might understand the material presented without the need for consulting other texts. Moreover, we have not tried to cover all aspects of the field, preferring to concentrate on the fundamental concepts. Nevertheless, the two volumes should prove useful as a reference guide to researchers just starting out in the field.

To fully familiarize oneself with the subject, it is important to practice solving problems. The book contains a large number of exercises (with solutions), which are an essential complement to the main text. In order to develop a solid understanding of the arguments dealt with here, it is indispensable that the student try to solve a large part of them.

Note to the reader

Some of the material presented is not necessary for understanding the rest of the book and may be omitted on a first reading. We have adopted two methods of highlighting such parts:

1) The sections or subsections with an asterisk before the title contain more advanced or complementary material. Such parts may be omitted without risk of encountering problems in reading the rest of the book.

2) Comments, notes or examples are printed in a small typeface.

Acknowledgments

We are indebted to several colleagues for criticism and suggestions. In particular, we wish to thank Alberto Bertoni, Gabriel Carlo, Rosario Fazio, Bertrand Georgeot, Luigi Lugiato, Sandro Morasca, Simone Montangero, Massimo Palma, Saverio Pascazio, Nicoletta Sabadini, Marcos Saraceno, Stefano Serra Capizzano and Robert Walters, who read preliminary versions of the book. Special thanks is due to Philip Ratcliffe, for useful remarks and suggestions, which substantially improved our book. Obviously no responsibility should be attributed to any of the above regarding possible flaws that might remain, for which the authors alone are to blame.

Contents

Contents of Volume II

Introduction

Quantum mechanics has had an enormous technological and societal impact. To appreciate this point, it is sufficient to consider the invention of the transistor, perhaps the most remarkable among the countless other applications of quantum mechanics. On the other hand, it is also easy to see the enormous impact of computers on everyday life. The importance of computers is such that it is appropriate to say that we are now living in the *information age*. This information revolution became possible thanks to the invention of the transistor, that is, thanks to the synergy between computer science and quantum physics.

Today this synergy offers completely new opportunities and promises exciting advances in both fundamental science and technological application. We are referring here to the fact that **quantum mechanics can be used to process and transmit information**.

Miniaturization provides us with an intuitive way of understanding why, in the near future, quantum laws will become important for computation. The electronics industry for computers grows hand-in-hand with the decrease in size of integrated circuits. This miniaturization is necessary to increase computational power, that is, the number of floating-point operations per second (flops) a computer can perform. In the 1950's, electronic computers based on vacuum-tube technology were capable of performing approximately 10^3 floating-point operations per second, while nowadays there exist supercomputers whose power is greater than 10 teraflops (10^{13} flops). As we have already remarked, this enormous growth of computational power has been made possible owing to progress in miniaturization, which may be quantified empirically in Moore's law. This law is the result of a remarkable observation made by Gordon Moore in 1965: the number

of transistors that may be placed on a single integrated-circuit chip doubles approximately every $18 - 24$ months. This exponential growth has not yet saturated and Moore's law is still valid. At the present time the limit is approximately 10^8 transistors per chip and the typical size of circuit components is of the order of 100 nanometres. Extrapolating Moore's law, one would estimate that around the year 2020 we shall reach the atomic size for storing a single bit of information. At that point, quantum effects will become unavoidably dominant.

It is clear that, besides quantum effects, other factors could bring Moore's law to an end. In the first place, there are economic considerations. Indeed, the cost of building fabrication facilities to manufacture chips has also increased exponentially with time. Nevertheless, it is important to understand the ultimate limitations set by quantum mechanics. Even though we might overcome economic barriers by means of technological breakthroughs, quantum physics sets fundamental limitations on the size of the circuit components. The first question under debate is whether it would be more convenient to push the silicon-based transistor to its physical limits or instead to develop alternative devices, such as quantum dots, single-electron transistors or molecular switches. A common feature of all these devices is that they are on the nanometre length scale and therefore quantum effects play a crucial role.

So far, we have talked about quantum switches that could substitute silicon-based transistors and possibly be connected together to execute classical algorithms based on Boolean logic. In this perspective, quantum effects are simply unavoidable corrections that must be taken into account owing to the nanometre size of the switches. A quantum computer represents a radically different challenge: the aim is to build a machine *based on quantum logic*, that is, it processes the information and performs logic operations by exploiting the laws of quantum mechanics.

The unit of quantum information is known as a *qubit* (the quantum counterpart of the classical *bit*) and a quantum computer may be viewed as a many-qubit system. Physically, a qubit is a two-level system, like the two spin states of a spin-$\frac{1}{2}$ particle, the vertical and horizontal polarization states of a single photon or the ground and excited states of an atom. A quantum computer is a system of many qubits, whose evolution can be controlled, and a quantum computation is a unitary transformation that acts on the many-qubit state describing the quantum computer.

The power of quantum computers is due to typical quantum phenomena, such as the *superposition* of quantum states and *entanglement*. There is an

inherent quantum parallelism associated with the superposition principle. In simple terms, a quantum computer can process a large number of classical inputs in a single run. On the other hand, this implies a large number of possible outputs. It is the task of quantum algorithms, which are based on quantum logic, to exploit the inherent quantum parallelism of quantum mechanics to highlight the desired output. In short, to be useful, quantum computers require the development of appropriate quantum software, that is, of efficient quantum algorithms.

In the 1980's Feynman suggested that a quantum computer based on quantum logic would be ideal for simulating quantum-mechanical systems and his ideas have spawned an active area of research in physics. It is also remarkable that quantum mechanics can help in the solution of basic problems of computer science. In 1994, Peter Shor proposed a quantum algorithm that efficiently solves the prime-factorization problem: given a composite integer, find its prime factors. This is a central problem in computer science and it is conjectured, though not proven, that for a classical computer it is computationally difficult to find the prime factors. Shor's algorithm efficiently solves the integer factorization problem and therefore it provides an exponential improvement in speed with respect to any known classical algorithm. It is worth mentioning here that there are cryptographic systems, such as RSA, that are used extensively today and that are based on the conjecture that no efficient algorithms exist for solving the prime factorization problem. Hence, Shor's algorithm, if implemented on a large-scale quantum computer, would break the RSA cryptosystem. Lov Grover has shown that quantum mechanics can also be useful for solving the problem of searching for a marked item in an unstructured database. In this case, the gain with respect to classical computation is quadratic.

Another interesting aspect of the quantum computer is that, in principle, it avoids dissipation. Present day classical computers, which are based on irreversible logic operations (gates), are *intrinsically* dissipative. The minimum energy requirements for irreversible computation are set by Landauer's principle: each time a single bit of information is erased, the amount of energy dissipated into the environment is at least $k_B T \ln 2$, where k_B is Boltzmann's constant and T the temperature of the environment surrounding the computer. Each irreversible classical gate must dissipate at least this amount of energy (in practice, present-day computers dissipate more by orders of magnitude). In contrast, quantum evolution is unitary and thus quantum logic gates must be reversible. Therefore, at least in principle, there is no energy dissipation during a quantum computer run.

It is well known that a small set of elementary logic gates allows the implementation of any complex computation on a classical computer. This is very important: it means that, when we change the problem, we do not need to modify our computer hardware. Fortunately, the same property remains valid for a quantum computer. It turns out that, in the quantum circuit model, each unitary transformation acting on a many-qubit system can be decomposed into gates acting on a single qubit and a single gate acting on two qubits, for instance the CNOT gate.

A large number of different proposals to build real quantum computers have been put forward. They range from NMR quantum processors to cold ion traps, superconducting tunnel-junction circuits and spin in semiconductors, to name but a few. Even though in some cases elementary quantum gates have been realized and quantum algorithms with a small number of qubits demonstrated, it is too early to say what type of implementation will be the most suitable to build a scalable piece of quantum hardware. Although for some computational problems the quantum computer is more powerful than the classical computer, still we need 50–1000 qubits and from thousands to millions of quantum gates to perform tasks inaccessible to the classical computer (the exact numbers depend, of course, on the specific quantum algorithm).

The technological challenge of realizing a quantum computer is very demanding: we need to be able to control the evolution of a large number of qubits for the time necessary to perform many quantum gates. Decoherence may be considered the ultimate obstacle to the practical realization of a quantum computer. Here the term decoherence denotes the decay of the quantum information stored in a quantum computer, due to the inevitable interaction of the quantum computer with the environment. Such interaction affects the performance of a quantum computer, introducing errors into the computation. Another source of errors that must be taken into account is the presence of imperfections in the quantum-computer hardware. Even though quantum error-correcting codes exist, a necessary requirement for a successful correction procedure is that one can implement many quantum gates inside the decoherence time scale. Here "many" means 10^3–10^4, the exact value depending on the kind of error. It is very hard to fulfil this requirement in complex many-qubit quantum systems.

The following question then arises: is it possible to build a useful quantum computer that could outperform existing classical computers in important computational tasks? And, if so, when? Besides the problem of decoherence, we should also remark on the difficulty of finding new and

efficient quantum algorithms. We know that the integer-factoring problem can be solved efficiently on a quantum computer, but we do not know the answer to the following fundamental question: What class of problems could be simulated efficiently on a quantum computer? Quantum computers open up fascinating prospects, but it does not seem likely that they will become a reality with practical applications in a few years. How long might it take to develop the required technology? Even though unexpected technological breakthroughs are, in principle, always possible, one should remember the enormous effort that was necessary in order to develop the technology of classical computers.

Nevertheless, even the first, modest, demonstrative experiments are remarkable, as they allow for testing the theoretical principles of quantum mechanics. Since quantum mechanics is a particularly counter-intuitive theory, we should at the very least expect that experiments and theoretical studies on quantum computation will provide us with a better understanding of quantum mechanics. Moreover, such research stimulates the control of individual quantum systems (atoms, electrons, photons *etc.*). We stress that this is not a mere laboratory curiosity, but has interesting technological applications. For instance, it is now possible to realize single-ion clocks that are more precise than standard atomic clocks. In a sense we may say that quantum computation rationalizes the efforts of the various experiments that manipulate individual quantum systems.

Another important research direction concerns the (secure) transmission of information. In this case, quantum mechanics allows us to perform not only faster operations but also operations *inaccessible* to classical means. Entanglement is at the heart of many quantum-information protocols. It is the most spectacular and counter-intuitive manifestation of quantum mechanics, observed in composite quantum systems: it signifies the existence of non-local correlations between measurements performed on well-separated particles. After two classical systems have interacted, they are in well-defined individual states. In contrast, after two quantum particles have interacted, in general, they can no longer be described independently of each other. There will be purely quantum correlations between two such particles, independently of their spatial separation. This is the content of the celebrated EPR paradox, a *Gedanken* experiment proposed by Einstein, Podolsky and Rosen in 1935. These authors showed that quantum theory leads to a contradiction, provided that we accept the two, seemingly natural, principles of realism and locality. The reality principle states that,

if we can predict with certainty the value of a physical quantity, then this value has physical reality, independently of our observation. The locality principle states that, if two systems are causally disconnected, the results of any measurement performed on one system cannot influence the result of a measurement performed on the second system. In other words, information cannot travel faster than the speed of light.

In 1964 Bell proved that this point of view (known as local realism) leads to predictions, Bell's inequalities, that are in contrast with quantum theory. Aspect's experiments (1982), performed with pairs of entangled photons, exhibited an unambiguous violation of a Bell's inequality by tens of standard deviations and an impressive agreement with quantum mechanics. These experiments also showed that it is possible to perform laboratory investigations on the more fundamental, non-intuitive aspects of quantum theory. More recently, other experiments have come closer to the requirements of the ideal EPR scheme. More generally, thanks to the development and increasing precision of experimental techniques, *Gedanken* experiments of the past become present-day real experiments.

The profound significance of Bell's inequalities and Aspect's experiments lies far beyond that of a mere consistency test of quantum mechanics. These results show that entanglement is a fundamentally new resource, beyond the realm of classical physics, and that it is possible to experimentally manipulate entangled states.

Quantum entanglement is central to many quantum-communication protocols. Of particular importance are *quantum dense coding*, which permits transmission of two bits of classical information through the manipulation of only one of two entangled qubits, and *quantum teleportation*, which allows the transfer of the state of one quantum system to another over an arbitrary distance. In recent experiments, based on photon pairs, entanglement has been distributed with the use of optical-fibre links, over distances of up to 10 kilometres. The long-distance free-space distribution of entanglement has also been recently demonstrated, with the two receivers of the entangled photons being separated by 600 metres. It is important to point out that the turbulence encountered along such an optical path is comparable to the effective turbulence in an earth-to-satellite transmission. Therefore, one may expect that in the near future it will become possible to distribute entanglement between receivers located very far apart (in two different continents, say) using satellite-based links.

Quantum mechanics also provides a unique contribution to cryptography: it enables two communicating parties to detect whether the transmit-

ted message has been intercepted by an eavesdropper. This is not possible in the realm of classical physics as it is always possible, in principle, to copy classical information without changing the original message. In contrast, in quantum mechanics the measurement process, in general, disturbs the system for fundamental reasons. Put plainly, this is a consequence of the Heisenberg uncertainty principle. Experimental advances in the field of quantum cryptography are impressive and quantum-cryptographic protocols have been demonstrated, using optical fibres, over distances of a few tens of kilometres at rates of the order of a thousand bits per second. Furthermore, free-space quantum cryptography has been demonstrated over distances up to several kilometres. In the near future, therefore, quantum cryptography could well be the first quantum-information protocol to find commercial applications.

To conclude this introduction, let us quote Schrödinger [*Brit. J. Phil. Sci.*, **3**, 233 (1952)]: "*We never experiment with just one electron or atom or (small) molecule. In thought-experiments we sometimes assume that we do; this invariably entails ridiculous consequences ... we are not experimenting with single particles, any more than we can raise Ichthyosauria in the zoo.*" It is absolutely remarkable that only fifty years later experiments on single electrons, atoms and molecules are routinely performed in laboratories all over the world.

A guide to the bibliography

We shall conclude each chapter with a short guide to the bibliography. Our aim is to give general references that might be used by the reader as an entry point for a more in-depth analysis of the topics discussed in this book. We shall therefore often refer to review papers instead of the original articles.

General references on quantum information and computation are the lecture notes of Preskill (1998) and the books of Gruska (1999) and Nielsen and Chuang (2000). Introductory level texts include Williams and Clearwater (1997), Pittenger (2000) and Hirvensalo (2001). Useful lecture notes have been prepared by Aharonov (2001), Vazirani (2002) and Mermin (2003). Mathematical aspects of quantum computation are discussed in Brylinski and Chen (2002). Interesting collections of review papers are to be found in Lo *et al.* (1998), Alber *et al.* (2001), Lomonaco (2002) and Bouwmeester *et al.* (2000). This last text is particularly interesting from

the point of view of experimental implementations.

Useful review papers of quantum computation and information are those of Steane (1998) and Galindo and Martin-Delgado (2002). The basic concepts of quantum computation are discussed in Ekert *et al.* (2001). A very readable review article of quantum information and computation is due to Bennett and DiVincenzo (2000).

A bibliographic guide containing more than 8000 items (updated as of June 2003) on the foundations of quantum mechanics and quantum information can be found in Cabello (2000–2003).

Chapter 1

Introduction to Classical Computation

This chapter introduces the basic concepts of computer science that are necessary for an understanding of quantum computation and information. We discuss the Turing machine, the fundamental model of computation since it formalizes the intuitive notion of an algorithm: if there exists an algorithm to solve a given problem, then this algorithm can be run on a Turing machine. We then introduce the circuit model of computation, which is equivalent to the Turing machine model but is nearer to real computers. In this model, the information is carried by wires and a small set of elementary logical operations (gates) allows implementation of any complex computation. It is important to find the minimum resources (computer memory, time and energy) required to solve a given problem with the best possible algorithm. This is the task of computational complexity, for which we provide a quick glance at the key concepts. Finally, we examine the energy resources necessary to perform computations. Here we discuss the relation between energy and information, which was explained by Landauer and Bennett in their solution of Maxwell's demon paradox. In particular, Landauer's principle sets the minimum energy requirements for irreversible computation. On the other hand, it turns out that it is, in principle, possible to perform any complex computation by means of reversible gates, without energy dissipation. A concrete model of reversible computation, the so-called billiard-ball computer, is briefly discussed.

1.1 The Turing machine

An *algorithm* is a set of instructions for solving a given problem. Examples of algorithms are those learnt at primary schools for adding and multiplying two integer numbers. Such algorithms always give the correct result when

applied to any pair of integer numbers.

The *Turing machine*, introduced by the mathematician Alan Turing in the 1930's, provides a precise mathematical formulation of the intuitive concept of algorithm. This machine contains the essential elements (memory, control unit and read/write unit) on which any modern computer is based. Turing's work was stimulated by an intense debate at that time regarding the following question: for which class or classes of problems is it possible to find an algorithm? This debate was motivated by a profound question raised by David Hilbert at the beginning of the twentieth century. Hilbert asked whether or not an algorithm might exist that could, in principle, be used to solve all mathematical problems. Hilbert thought (erroneously, as we shall see in this section) that the answer to his question was positive.

A closely related problem is the following: given a logical system defined by an ensemble of axioms and rules, can all possible propositions be proved, at least in principle, to be either true or false? At the beginning of twentieth century it was widely believed that the answer to this question was also positive. (Of course, the question does not address the problem that, in practice, it may be extremely difficult to prove whether a proposition is true or false). Contrary to this belief, in the 1930's Kurt Gödel proved a theorem stating that there exist mathematical propositions of any given logical system that are *undecidable*, meaning that they can neither be proved nor disproved using axioms and rules inside the same logical system. This does not exclude that we can enlarge the system, introducing new axioms and rules, and thus decide whether a given proposition is true or false. However, it will also be possible to find undecidable propositions in this new system. Thus, it turns out that logical systems are intrinsically *incomplete*. Notice that Gödel's theorem also sets limits on the possibilities of a computer: it cannot answer all questions on arithmetics.

The main elements of a Turing machine are illustrated in Fig. 1.1. The general idea is that the machine performs a computation as a "human computer" would. Such a human computer is capable of storing only a limited amount of information in his brain, but has at his disposal an (ideally) unlimited amount of paper for reading and writing operations. Likewise, the Turing machine contains the following three main elements:

1. A *tape*, which is infinite and divided into cells. Each cell holds only one letter a_i from a finite alphabet $\{a_1, a_2, \ldots, a_k\}$ or is blank. Except for a finite number of cells, all the other cells are blank.
2. A *control unit*, which has a finite number of states, $\{s_1, s_2, \ldots, s_l, H\}$,

where H is a special state, known as the halting state: if the state of the control unit becomes H, then the computation terminates.

3. A *read/write head*, which addresses a single cell of the tape. It reads and (over)writes or erases a letter in this cell, after which, the head moves one cell to the left or to the right.

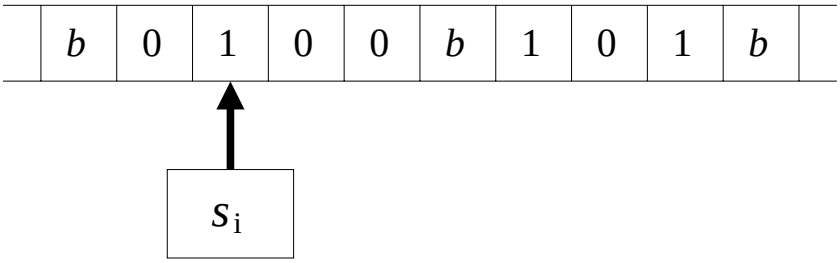

Fig. 1.1 Schematic drawing of a Turing machine. The symbol b denotes a blank cell.

The working of a Turing machine is governed by a *program*, which is simply a finite set of instructions. Each instruction governs one step of the Turing machine and induces the following sequence of operations:

(i) the transition of the control unit from the state s to the state $\overline{s}$,
(ii) the transition of the cell addressed by the read/write head from the letter a to the letter $\overline{a}$,
(iii) the displacement of the read/write head one cell left or right.

Therefore, an instruction in the Turing machine is defined by three functions f_S, f_A and f_D, defined as follows:

$$\overline{s} = f_S(s,a)\,, \tag{1.1a}$$
$$\overline{a} = f_A(s,a)\,, \tag{1.1b}$$
$$d = f_D(s,a)\,, \tag{1.1c}$$

where d indicates the displacement of the head to the left ($d = l$) or to the right ($d = r$). In short, the functions f_S, f_A and f_D define the mapping

$$(s,a) \;\to\; (\overline{s},\overline{a},d)\,. \tag{1.2}$$

1.1.1 *Addition on a Turing machine*

Let us now describe a concrete example: a Turing machine performing the addition of two integers. For the sake of simplicity, we write the integer numbers in the unary representation: an integer N is written as a sequence of N 1's, that is, $1 = 1$, $2 = 11$, $3 = 111$, $4 = 1111$ and so on. As an example, we compute the sum $2 + 3$. Our Turing machine needs five internal states $\{s_1, s_2, s_3, s_4, H\}$ and a unary alphabet, namely, the single letter 1. We denote by b the blank cells of the tape. The initial condition of the machine is shown in Fig. 1.2: the initial state is s_1, the head points to a well-defined cell and the numbers to be added, $2 = 11$ and $3 = 111$, are written on the tape, separated by a blank.

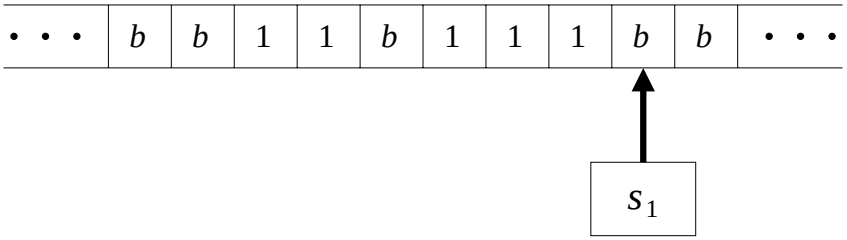

Fig. 1.2 Initial conditions of a Turing machine for computing the sum $2 + 3$.

The program for computing the sum of two integer numbers is shown in Table 1.1. The program has a total number of six lines. The internal state

Table 1.1 The algorithm for computing the sum of two integers on a Turing machine.

s	a	$\overline{s}$	$\overline{a}$	d
s_1	b	s_2	b	l
s_2	b	s_3	b	l
s_2	1	s_2	1	l
s_3	b	H	b	0
s_3	1	s_4	b	r
s_4	b	s_2	1	l

s of the machine and the letter a being read on the tape determine which program line is executed. The last three columns of Table 1.1 denote the new state $\overline{s}$, the letter $\overline{a}$ overwritten on the tape and the left/right direction ($d = l$ or $d = r$) of the read/write head motion. Note that in the fourth

line of the program $d = 0$ since the machine halts and the head moves no further. It is easy to check that, if we start from the initial conditions of Fig. 1.2 and run the program of Table 1.1, the machine halts in the configuration depicted in Fig. 1.3 and we can read on the tape the result of the sum, $2 + 3 = 5$. It is also easy to convince ourselves that the same program can compute the sum of two arbitrary integers m and n, provided that the initial conditions are set as in Fig. 1.4.

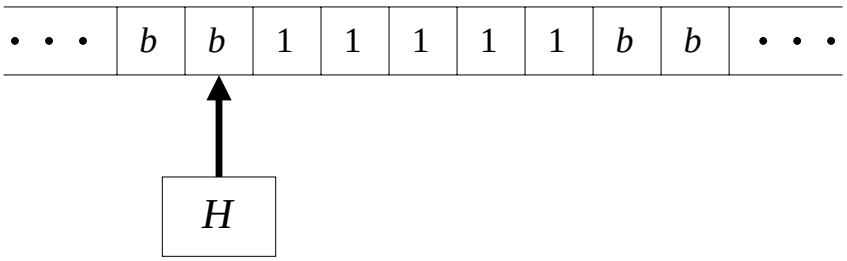

Fig. 1.3 A Turing machine after computation of the sum $2 + 3$. The machine started from the initial conditions of Fig. 1.2 and implemented the program of Table 1.1.

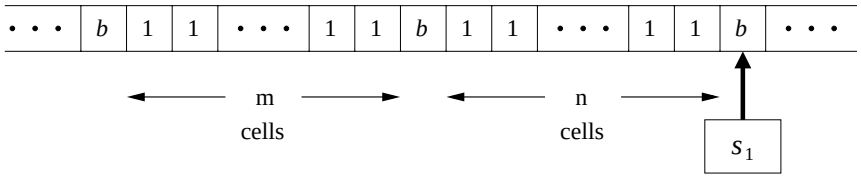

Fig. 1.4 The initial conditions of a Turing machine for computing the sum $m + n$ of two generic integers.

1.1.2 *The Church–Turing thesis*

It turns out that Turing machines are capable of solving very complex problems. As far as we know, they can be used to simulate any operation carried out on a modern computer. If there exists an algorithm to compute a function, then the computation can be performed by a Turing machine. This idea was formalized independently by Church and Turing:

The Church–Turing thesis: *The class of all functions computable by a Turing machine is equivalent to the class of all functions computable by means of an algorithm.*