Computer Physics Communications 181 (2010) 283–300

Contents lists available at ScienceDirect

Computer Physics Communications

www.elsevier.com/locate/cpc

Quantum computer simulation using the CUDA programming model

Eladio Gutiérrez *, Sergio Romero, María A. Trenas, Emilio L. Zapata

Department of Computer Architecture, University of Málaga, 29071 Málaga, Spain

ARTICLE INFO

Article history:
Received 25 November 2008
Received in revised form 1 September 2009
Accepted 22 September 2009
Available online 30 September 2009

PACS:
03.67.Ac

Keywords:
Quantum computing
Parallel processing
Graphics Processing Unit (GPU)
Computer Unified Device Architecture (CUDA)
Single Instruction Multiple Threads (SIMT)

ABSTRACT

Quantum computing emerges as a field that captures a great theoretical interest. Its simulation represents a problem with high memory and computational requirements which makes advisable the use of parallel platforms. In this work we deal with the simulation of an ideal quantum computer on the Compute Unified Device Architecture (CUDA), as such a problem can benefit from the high computational capacities of Graphics Processing Units (GPU). After all, modern GPUs are becoming very powerful computational architectures which is causing a growing interest in their application for general purpose. CUDA provides an execution model oriented towards a more general exploitation of the GPU allowing to use it as a massively parallel SIMT (Single-Instruction Multiple-Thread) multiprocessor. A simulator that takes into account memory reference locality issues is proposed, showing that the challenge of achieving a high performance depends strongly on the explicit exploitation of memory hierarchy. Several strategies have been experimentally evaluated obtaining good performance results in comparison with conventional platforms.

1. Introduction

Not even two decades have barely elapsed since the becoming of quantum computation as a promising discipline. When compared with conventional computation (classic), the so denominated quantum computers are devices that process information on the basis of the laws of the quantum physics and that could solve some problems of nonpolynomic complexity in a much smaller time [14]. Although this is a quite hopeful approach, at the present time certain hard limitations must still be faced. On the one hand, existing technology only allows to construct quantum computers of very reduced dimensions [9], and on the other hand only a very small number of algorithms are known to diminish significantly their complexity when executed in a quantum computer, as they are [6,10,21].

The interest of simulating the behavior of quantum computers is motivated by several factors that go from the purely theoretical, academic and educational ones, to the benchmarking of conventional mono or multiprocessor platforms. Besides, its simulation constitutes a very useful tool in the development and test of new quantum algorithms (at least with data sets of small dimension).

Most of the power of quantum computers is due to the quantum parallelism that allows to perform simultaneous operations on an exponential set of superimposed data. This is why the simulation of this kind of systems requires an exponential effort. Parallelism is a suitable tool in order to mitigate such computational requirements and will allow the emulation of quantum computers of a greater dimension in a reasonable time. The model shown in Fig. 1 is the one followed in this work [18]. The quantum computer acts like an accelerating hardware of the classic processor, which sends the orders required to solve a concrete problem. According to the laws that govern it, it is not possible to know the exact state of this quantum computer. Therefore, the output of the quantum algorithm will be obtained by a process of measurement with certain probability. Such an approach is followed in our parallel simulator, which considers an ideal quantum computer. The interface adopted between the classic computer (host) and the simulation platform is borrowed from libquantum [4], one of the more popular simulation software.

In this paper we show that quantum computers can be efficiently simulated on modern architectures based on Graphic Processing Units (GPU). With this purpose, we propose several approaches to the parallel simulation of the basic operators of an ideal quantum computer framed within the Compute Unified Device Architecture (CUDA) after NVIDIA [16].

2. Related work

The interest of scientists, engineers and academics in quantum computing has given rise to the development of several quantum computer simulators, of which a wide list can be found at [19].

* Corresponding author.
E-mail addresses: eladio@uma.es (E. Gutiérrez), sromero@uma.es (S. Romero), matrenas@uma.es (M.A. Trenas), ezapata@ac.uma.es (E.L. Zapata).

0010-4655/$ – see front matter
doi:10.1016/j.cpc.2009.09.021

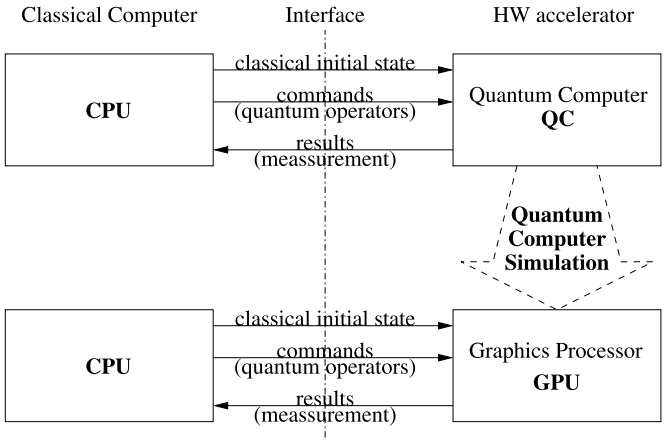

Fig. 1. Quantum computer model as a hardware accelerator and its simulation.

They cover a large number of programming languages and most of them are oriented to instructional or demonstration purposes. Among them we can highlight libquantum [4] a C-based library that allows to write "quantum programs" making use of function calls to elementary quantum gates. The libquantum is a popular library distributed in source format and whose binaries are available in some Linux distributions. In addition it takes part of the benchmark SPEC CPU2006 [22].

Efforts to cope with the high memory and computational requirements can be found in literature. As hardware solutions, several systems based on FPGAs has been proposed as hardware accelerators for mayor quantum computing algorithms, like [1,8,12, 13]. On the software side, massive parallel platforms provide high computational resources in order to run faster simulations for a high number of qubits. The main challenge to overcome is how to distribute/communicate efficiently data among processors.

In [17] parallel simulations were carried out on a CrayT3E and an IBM SP2 with MPI message passing library in C, reaching up to 4096 processors with a maximum of 28 qubits. In [15], parallel simulations were executed on a UltraSparcII-based SunEnterprise 4500, using the Solaris thread library, reaching up to 8 threads with a maximum size of 30 qubits. Parallel simulations in [9] were carried out on a Beowulf cluster of 16 nodes based on Pentium 4 processors by using a message passing model allowing to run simulations up to 29 qubits. Simulations proposed in [2] run over an IBM P690 cluster by means of a hybrid model OpenMP (shared memory)/MPI (message passing) with a maximum of 1024 threads achieving executions up to 37 qubits. More recently, in [20], several simulations are tested and compared on massively parallel platforms including IBM Blue Gene/L, IBM Regatta P690, Hitachi SR11000, Cray X1E and others. Programs were coded in Fortran 90 with the MPI library, providing executions with 4096 threads with sizes up to 36 qubits.

3. Quantum computing fundamentals

The ideal quantum computer to be simulated follows the model presented in [5], consisting on the successive application of quantum gates (gate network) to a register with a classical initial state. The quantum bit (qubit) can be imagined as the linear superposition of two homologous classical states denoted as $|0\rangle$, $|1\rangle$, in Dirac notation. Its state can be represented using a complex two-dimensional vector by $\Psi = \alpha_0|0\rangle + \alpha_1|1\rangle$, where the coefficients, or amplitudes, verify $|\alpha_0|^2 + |\alpha_1|^2 = 1$. $|\alpha_0|^2$ and $|\alpha_1|^2$ are interpreted as the probability of measuring $|0\rangle$ or $|1\rangle$. In vector notation, $\Psi = \binom{\alpha_1}{\alpha_0}$, being $|0\rangle = \binom{0}{1}$, $|1\rangle = \binom{1}{0}$.

A quantum register generalizes the qubit definition. The state of an n-qubit quantum register is determined by the linear superposition of the 2^n possible classical states provided by n bits. After this, the state of a quantum register can be written as $\Psi = \sum_{i=0}^{2^n-1} \alpha_i|i\rangle$ with $\alpha_i \in \mathbb{C}$, $\sum_{i=0}^{2^n-1} |\alpha_i|^2 = 1$, where $|\alpha_i|^2$ is interpreted as the probability of obtaining $|i\rangle$ when the register is measured.

Let Ψ be an element of a 2^n-dimensional complex vector space, where $|i\rangle$ constitutes a basis, with $0 \leqslant i \leqslant 2^n - 1$. For example, for $n = 3$, we will write $|6\rangle = |110\rangle = (0\ 0\ 0\ 0\ 0\ 0\ 1\ 0)^T$. By using the Kronecker's tensor product, the elements of the state space basis can be represented by a function of the individual states of the qubits. For example $|6\rangle = |110\rangle = |1\rangle \otimes |1\rangle \otimes |0\rangle$.

The state of a quantum register will evolve according to a transformation, which can be interpreted as an operator U applied to the register state. Quantum physics laws settle that operator U must be a linear and unitary one. It follows that for an n-qubit register, a $(2^n \times 2^n)$-size matrix can be found verifying $UU^* = I$, where U^* is matrix U conjugated and transposed, and I is the unitary matrix. As a consequence, every valid transformation must be reversible. Usually, this kind of transformations are represented as in Fig. 2(a).

Let us consider the application of a transformation over one particular bit, (Fig. 2(b)). In this case, the global transformation will be the tensor product of all the 1-qubit transformations simultaneously applied to each individual qubit. If the 1-qubit operator U is applied to the k-th qubit then:

$$U_g = I \otimes \overset{(n-k-1 \text{ times})}{\cdots} \otimes I \otimes U \otimes I \otimes \overset{(k \text{ times})}{\cdots} \otimes I = I^{\otimes n-k-1} \otimes U \otimes I^{\otimes k}. \quad (1)$$

The transformation applied to one single qubit is given by an unitary quantum gate of order 2×2. Table 1 presents several well-known transformations. The generalization to gates with more than one qubit is straightforward, resulting in an associated matrix of order $2^n \times 2^n$, for n qubits. Notice that the number of qubits at the gate's output must be equal to the one at its input, as it is a reversible transformation. This does not occur with conventional logic. Different minimal universal sets of gates have been proposed, in such a way that any n-qubit transformation can be expressed as a network of these gates. It is proven that no universal minimal set of 1-qubit gates exists. Nevertheless a complete set

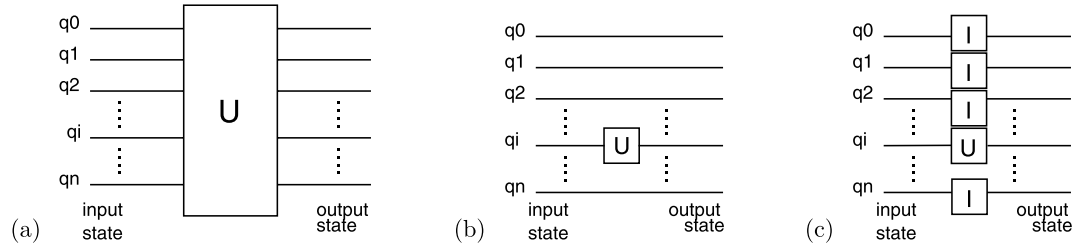

Fig. 2. Representation of quantum transformations as quantum gate networks.