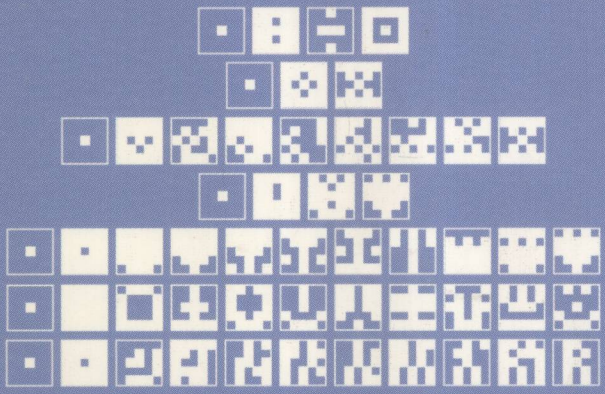
Jean-Pierre Banâtre
Pascal Fradet
Jean-Louis Giavitto
Olivier Michel (Eds.)

Hot Topics
LNCS 3566

Unconventional Programming Paradigms

International Workshop UPP 2004
Le Mont Saint Michel, France, September 2004
Revised Selected and Invited Papers



Springer

Jean-Pierre Banâtre Pascal Fradet
Jean-Louis Giavitto Olivier Michel (Eds.)

Unconventional Programming Paradigms

International Workshop UPP 2004
Le Mont Saint Michel, France, September 15-17, 2004
Revised Selected and Invited Papers

Springer

Volume Editors

Jean-Pierre Banâtre
Université de Rennes I and INRIA/IRISA
Campus de Beaulieu, 35042 Rennes Cedex, France
E-mail: jpbanatre@inria.fr

Pascal Fradet
INRIA Rhône-Alpes
655 av. de l'Europe, 38330 Montbonnot, France
E-mail: Pascal.Fradet@inria.fr

Jean-Louis Giavitto
LaMI/Université d'Évry Val d'Essonne
Tour Evry 2, GENOPOLE, 523 Place des terrasses de l'agora, 91000 Évry, France
E-mail: giavitto@lami.univ-evry.fr

Olivier Michel
LaMI/Université d'Évry Val d'Essonne
Cours Monseigneur Romero, 91025 Evry Cedex, France
E-mail: michel@lami.univ-evry.fr

Library of Congress Control Number: 2005928846

CR Subject Classification (1998): F.1, D.1, D.3, F.3, F.4

ISSN 0302-9743
ISBN-10 3-540-27884-2 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-27884-9 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 11527800 06/3142 5 4 3 2 1 0

Lecture Notes in Computer Science 3566

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

Preface

Nowadays, developers have to face the proliferation of hardware and software environments, the increasing demands of the users, the growing number of programs and the sharing of information, competences and services thanks to the generalization of data bases and communication networks. A program is no more a monolithic entity conceived, produced and finalized before being used. A program is now seen as an open and adaptive frame, which, for example, can dynamically incorporate services not foreseen by the initial designer. These new needs call for new control structures and program interactions.

Unconventional approaches to programming have long been developed in various niches and constitute a reservoir of alternative ways to face the programming languages crisis. New models of programming (e.g., bio-inspired computing, artificial chemistry, amorphous computing, ...) are also currently experiencing a renewed period of growth as they face specific needs and new application domains. These approaches provide new abstractions and notations or develop new ways of interacting with programs. They are implemented by embedding new sophisticated data structures in a classical programming model (API), by extending an existing language with new constructs (to handle concurrency, exceptions, open environments, ...), by conceiving new software life cycles and program executions (aspect weaving, run-time compilation) or by relying on an entire new paradigm to specify a computation. They are inspired by theoretical considerations (e.g., topological, algebraic or logical foundations), driven by the domain at hand (domain-specific languages like PostScript, musical notation, animation, signal processing, etc.) or by metaphors taken from various areas (quantum computing, computing with molecules, information processing in biological tissues, problem solving from nature, ethological and social modeling). The practical applications of these new programming paradigms and languages prompt research into the expressivity, semantics and implementation of programming languages and systems architectures, as well as into the algorithmic complexity and optimization of programs.

The purpose of the workshop was to bring together researchers from the various communities working on wild and crazy ideas in programming languages to present their results, to foster fertilization between theory and practice, and to favor the dissemination and growth of new programming paradigms.

The contributions were split up into five tracks:

- Chemical Computing
- Amorphous Computing
- Bio-inspired Computing
- Autonomic Computing
- Generative Programming

This workshop kept the same informal style of a previous successful meeting held in 1991 in Le Mont Saint Michel under the title *New Directions in High-Level Parallel Programming Languages*. Each track was handled by a well-known researcher in the concerned area. Each track leader was in charge of inviting other researchers on his topic and organizing his session. These track leaders plus the four promoters of this initiative constituted the Program Committee of the workshop (see below). This volume gathers extended and revised versions of most of the papers presented at the workshop, including the invited presentation given by Philippe Jorrand on quantum computing.

On the practical side, several persons contributed to the success of the workshop. We offer our sincere thanks to all of them. We are particularly grateful to Edith Corre and Elisabeth Lebret of IRISA and to Rémi Ronchaud from ERCIM who were very efficient and professional in the organization. Finally, we address our sincere acknowledgments to all the participants who, beside the high quality of their scientific contribution, made the workshop a friendly and unique event.

April 2005

Jean-Pierre Banâtre
Pascal Fradet
Jean-Louis Giavitto
Olivier Michel

Organization

The workshop was jointly supported by the European Commission's Information Society Technologies Programme, Future and Emerging Technologies Activity, and the US National Science Foundation, Directorate for Computer and Information Science and Engineering. This workshop is part of a series of strategic workshops that identify key research challenges and opportunities in information technology. It was organized by ERCIM (European Research Consortium for Informatics and Mathematics) and received additional support from INRIA, Université d'Evry Val d'Essonne, Université de Rennes 1, and Microsoft Research.

Program Committee

Organizing Committee

Jean-Pierre Banâtre	Université de Rennes 1, and INRIA/IRISA, France
Pascal Fradet	INRIA Rhône-Alpes, France
Jean-Louis Giavitto	LaMI/Université d'Evry Val d'Essonne, France
Olivier Michel	LaMI/Université d'Evry Val d'Essonne, France

Track Leaders

Pierre Cointe	Ecole des Mines de Nantes, France *Generative Programming*
Daniel Coore	University of West Indies, Jamaica *Amorphous Computing*
Peter Dittrich	Friedrich Schiller University Jena, Germany *Chemical Computing*
Manish Parashar	Rutgers, The State University of New Jersey, USA *Autonomic Computing*
Gheorghe Păun	Institute of Mathematics of the Romanian Academy, Romania *Bio-inspired Computing*

Lecture Notes in Computer Science

For information about Vols. 1–3492

please contact your bookseller or Springer

Vol. 3626: B. Ganter, G. Stumme, R. Wille (Eds.), Formal Concept Analysis. X, 349 pages. 2005. (Subseries LNAI).

Vol. 3596: F. Dau, M.-L. Mugnier, G. Stumme (Eds.), Conceptual Structures: Common Semantics for Sharing Knowledge. XI, 467 pages. 2005. (Subseries LNAI).

Vol. 3587: P. Perner, A. Imiya (Eds.), Machine Learning and Data Mining in Pattern Recognition. XVII, 695 pages. 2005. (Subseries LNAI).

Vol. 3582: J. Fitzgerald, I.J. Hayes, A. Tarlecki (Eds.), FM 2005: Formal Methods. XIV, 558 pages. 2005.

Vol. 3580: L. Caires, G.F. Italiano, L. Monteiro, C. Palamidessi, M. Yung (Eds.), Automata, Languages and Programming. XXV, 1477 pages. 2005.

Vol. 3578: M. Gallagher, J. Hogan, F. Maire (Eds.), Intelligent Data Engineering and Automated Learning - IDEAL 2005. XVI, 599 pages. 2005.

Vol. 3576: K. Etessami, S.K. Rajamani (Eds.), Computer Aided Verification. XV, 564 pages. 2005.

Vol. 3575: S. Wermter, G. Palm, M. Elshaw (Eds.), Biomimetic Neural Learning for Intelligent Robots. IX, 383 pages. 2005. (Subseries LNAI).

Vol. 3574: C. Boyd, J.M. González Nieto (Eds.), Information Security and Privacy. XIII, 586 pages. 2005.

Vol. 3573: S. Etalle (Ed.), Logic Based Program Synthesis and Transformation. VIII, 279 pages. 2005.

Vol. 3572: C. De Felice, A. Restivo (Eds.), Developments in Language Theory. XI, 409 pages. 2005.

Vol. 3571: L. Godo (Ed.), Symbolic and Quantitative Approaches to Reasoning with Uncertainty. XVI, 1028 pages. 2005. (Subseries LNAI).

Vol. 3570: A. S. Patrick, M. Yung (Eds.), Financial Cryptography and Data Security. XII, 376 pages. 2005.

Vol. 3569: F. Bacchus, T. Walsh (Eds.), Theory and Applications of Satisfiability Testing. XII, 492 pages. 2005.

Vol. 3568: W.K. Leow, M.S. Lew, T.-S. Chua, W.-Y. Ma, L. Chaisorn, E.M. Bakker (Eds.), Image and Video Retrieval. XVII, 672 pages. 2005.

Vol. 3567: M. Jackson, D. Nelson, S. Stirk (Eds.), Database: Enterprise, Skills and Innovation. XII, 185 pages. 2005.

Vol. 3566: J.-P. Banâtre, P. Fradet, J.-L. Giavitto, O. Michel (Eds.), Unconventional Programming Paradigms. XI, 367 pages. 2005.

Vol. 3565: G.E. Christensen, M. Sonka (Eds.), Information Processing in Medical Imaging. XXI, 777 pages. 2005.

Vol. 3564: N. Eisinger, J. Małuszyński (Eds.), Reasoning Web. IX, 319 pages. 2005.

Vol. 3562: J. Mira, J.R. Álvarez (Eds.), Artificial Intelligence and Knowledge Engineering Applications: A Bioinspired Approach, Part II. XXIV, 636 pages. 2005.

Vol. 3561: J. Mira, J.R. Álvarez (Eds.), Mechanisms, Symbols, and Models Underlying Cognition, Part I. XXIV, 532 pages. 2005.

Vol. 3560: V.K. Prasanna, S. Iyengar, P.G. Spirakis, M. Welsh (Eds.), Distributed Computing in Sensor Systems. XV, 423 pages. 2005.

Vol. 3559: P. Auer, R. Meir (Eds.), Learning Theory. XI, 692 pages. 2005. (Subseries LNAI).

Vol. 3558: V. Torra, Y. Narukawa, S. Miyamoto (Eds.), Modeling Decisions for Artificial Intelligence. XII, 470 pages. 2005. (Subseries LNAI).

Vol. 3557: H. Gilbert, H. Handschuh (Eds.), Fast Software Encryption. XI, 443 pages. 2005.

Vol. 3556: H. Baumeister, M. Marchesi, M. Holcombe (Eds.), Extreme Programming and Agile Processes in Software Engineering. XIV, 332 pages. 2005.

Vol. 3555: T. Vardanega, A. Wellings (Eds.), Reliable Software Technology – Ada-Europe 2005. XV, 273 pages. 2005.

Vol. 3554: A. Dey, B. Kokinov, D. Leake, R. Turner (Eds.), Modeling and Using Context. XIV, 572 pages. 2005. (Subseries LNAI).

Vol. 3553: T.D. Hämäläinen, A.D. Pimentel, J. Takala, S. Vassiliadis (Eds.), Embedded Computer Systems: Architectures, Modeling, and Simulation. XV, 476 pages. 2005.

Vol. 3552: H. de Meer, N. Bhatti (Eds.), Quality of Service – IWQoS 2005. XVIII, 400 pages. 2005.

Vol. 3551: T. Härder, W. Lehner (Eds.), Data Management in a Connected World. XIX, 371 pages. 2005.

Vol. 3548: K. Julisch, C. Kruegel (Eds.), Intrusion and Malware Detection and Vulnerability Assessment. X, 241 pages. 2005.

Vol. 3547: F. Bomarius, S. Komi-Sirviö (Eds.), Product Focused Software Process Improvement. XIII, 588 pages. 2005.

Vol. 3546: T. Kanade, A. Jain, N.K. Ratha (Eds.), Audio- and Video-Based Biometric Person Authentication. XX, 1134 pages. 2005.

Vol. 3543: L. Kutvonen, N. Alonistioti (Eds.), Distributed Applications and Interoperable Systems. XI, 235 pages. 2005.

Vol. 3542: H.H. Hoos, D.G. Mitchell (Eds.), Theory and Applications of Satisfiability Testing. XIII, 393 pages. 2005.

Vol. 3541: N.C. Oza, R. Polikar, J. Kittler, F. Roli (Eds.), Multiple Classifier Systems. XII, 430 pages. 2005.

Vol. 3540: H. Kalviainen, J. Parkkinen, A. Kaarna (Eds.), Image Analysis. XXII, 1270 pages. 2005.

Vol. 3537: A. Apostolico, M. Crochemore, K. Park (Eds.), Combinatorial Pattern Matching. XI, 444 pages. 2005.

Vol. 3536: G. Ciardo, P. Darondeau (Eds.), Applications and Theory of Petri Nets 2005. XI, 470 pages. 2005.

Vol. 3535: M. Steffen, G. Zavattaro (Eds.), Formal Methods for Open Object-Based Distributed Systems. X, 323 pages. 2005.

Vol. 3533: M. Ali, F. Esposito (Eds.), Innovations in Applied Artificial Intelligence. XX, 858 pages. 2005. (Subseries LNAI).

Vol. 3532: A. Gómez-Pérez, J. Euzenat (Eds.), The Semantic Web: Research and Applications. XV, 728 pages. 2005.

Vol. 3531: J. Ioannidis, A. Keromytis, M. Yung (Eds.), Applied Cryptography and Network Security. XI, 530 pages. 2005.

Vol. 3530: A. Prinz, R. Reed, J. Reed (Eds.), SDL 2005: Model Driven. XI, 361 pages. 2005.

Vol. 3528: P.S. Szczepaniak, J. Kacprzyk, A. Niewiadomski (Eds.), Advances in Web Intelligence. XVII, 513 pages. 2005. (Subseries LNAI).

Vol. 3527: R. Morrison, F. Oquendo (Eds.), Software Architecture. XII, 263 pages. 2005.

Vol. 3526: S.B. Cooper, B. Löwe, L. Torenvliet (Eds.), New Computational Paradigms. XVII, 574 pages. 2005.

Vol. 3525: A.E. Abdallah, C.B. Jones, J.W. Sanders (Eds.), Communicating Sequential Processes. XIV, 321 pages. 2005.

Vol. 3524: R. Barták, M. Milano (Eds.), Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems. XI, 320 pages. 2005.

Vol. 3523: J.S. Marques, N. Pérez de la Blanca, P. Pina (Eds.), Pattern Recognition and Image Analysis, Part II. XXVI, 733 pages. 2005.

Vol. 3522: J.S. Marques, N. Pérez de la Blanca, P. Pina (Eds.), Pattern Recognition and Image Analysis, Part I. XXVI, 703 pages. 2005.

Vol. 3521: N. Megiddo, Y. Xu, B. Zhu (Eds.), Algorithmic Applications in Management. XIII, 484 pages. 2005.

Vol. 3520: O. Pastor, J. Falcão e Cunha (Eds.), Advanced Information Systems Engineering. XVI, 584 pages. 2005.

Vol. 3519: H. Li, P. J. Olver, G. Sommer (Eds.), Computer Algebra and Geometric Algebra with Applications. IX, 449 pages. 2005.

Vol. 3518: T.B. Ho, D. Cheung, H. Liu (Eds.), Advances in Knowledge Discovery and Data Mining. XXI, 864 pages. 2005. (Subseries LNAI).

Vol. 3517: H.S. Baird, D.P. Lopresti (Eds.), Human Interactive Proofs. IX, 143 pages. 2005.

Vol. 3516: V.S. Sunderam, G.D.v. Albada, P.M.A. Sloot, J.J. Dongarra (Eds.), Computational Science – ICCS 2005, Part III. LXIII, 1143 pages. 2005.

Vol. 3515: V.S. Sunderam, G.D.v. Albada, P.M.A. Sloot, J.J. Dongarra (Eds.), Computational Science – ICCS 2005, Part II. LXIII, 1101 pages. 2005.

Vol. 3514: V.S. Sunderam, G.D.v. Albada, P.M.A. Sloot, J.J. Dongarra (Eds.), Computational Science – ICCS 2005, Part I. LXIII, 1089 pages. 2005.

Vol. 3513: A. Montoyo, R. Muñoz, E. Métais (Eds.), Natural Language Processing and Information Systems. XII, 408 pages. 2005.

Vol. 3512: J. Cabestany, A. Prieto, F. Sandoval (Eds.), Computational Intelligence and Bioinspired Systems. XXV, 1260 pages. 2005.

Vol. 3511: U.K. Wiil (Ed.), Metainformatics. VIII, 221 pages. 2005.

Vol. 3510: T. Braun, G. Carle, Y. Koucheryavy, V. Tsaoussidis (Eds.), Wired/Wireless Internet Communications. XIV, 366 pages. 2005.

Vol. 3509: M. Jünger, V. Kaibel (Eds.), Integer Programming and Combinatorial Optimization. XI, 484 pages. 2005.

Vol. 3508: P. Bresciani, P. Giorgini, B. Henderson-Sellers, G. Low, M. Winikoff (Eds.), Agent-Oriented Information Systems II. X, 227 pages. 2005. (Subseries LNAI).

Vol. 3507: F. Crestani, I. Ruthven (Eds.), Information Context: Nature, Impact, and Role. XIII, 253 pages. 2005.

Vol. 3506: C. Park, S. Chee (Eds.), Information Security and Cryptology – ICISC 2004. XIV, 490 pages. 2005.

Vol. 3505: V. Gorodetsky, J. Liu, V. A. Skormin (Eds.), Autonomous Intelligent Systems: Agents and Data Mining. XIII, 303 pages. 2005. (Subseries LNAI).

Vol. 3504: A.F. Frangi, P.I. Radeva, A. Santos, M. Hernandez (Eds.), Functional Imaging and Modeling of the Heart. XV, 489 pages. 2005.

Vol. 3503: S.E. Nikoletseas (Ed.), Experimental and Efficient Algorithms. XV, 624 pages. 2005.

Vol. 3502: F. Khendek, R. Dssouli (Eds.), Testing of Communicating Systems. X, 381 pages. 2005.

Vol. 3501: B. Kégl, G. Lapalme (Eds.), Advances in Artificial Intelligence. XV, 458 pages. 2005. (Subseries LNAI).

Vol. 3500: S. Miyano, J. Mesirov, S. Kasif, S. Istrail, P. Pevzner, M. Waterman (Eds.), Research in Computational Molecular Biology. XVII, 632 pages. 2005. (Subseries LNBI).

Vol. 3499: A. Pelc, M. Raynal (Eds.), Structural Information and Communication Complexity. X, 323 pages. 2005.

Vol. 3498: J. Wang, X. Liao, Z. Yi (Eds.), Advances in Neural Networks – ISNN 2005, Part III. XLIX, 1077 pages. 2005.

Vol. 3497: J. Wang, X. Liao, Z. Yi (Eds.), Advances in Neural Networks – ISNN 2005, Part II. XLIX, 947 pages. 2005.

Vol. 3496: J. Wang, X. Liao, Z. Yi (Eds.), Advances in Neural Networks – ISNN 2005, Part II. L, 1055 pages. 2005.

Vol. 3495: P. Kantor, G. Muresan, F. Roberts, D.D. Zeng, F.-Y. Wang, H. Chen, R.C. Merkle (Eds.), Intelligence and Security Informatics. XVIII, 674 pages. 2005.

Vol. 3494: R. Cramer (Ed.), Advances in Cryptology – EUROCRYPT 2005. XIV, 576 pages. 2005.

Vol. 3493: N. Fuhr, M. Lalmas, S. Malik, Z. Szlávik (Eds.), Advances in XML Information Retrieval. XI, 438 pages. 2005.

¥453.12元

Table of Contents

Invited Talk

Chemical Computing

Amorphous Computing

Bio-inspired Computing

Autonomic Computing

Generative Programming

From Quantum Physics to Programming Languages: A Process Algebraic Approach

Philippe Jorrand and Marie Lalire

Leibniz Laboratory, 46 avenue Felix Viallet,
38000 Grenoble, France
{Philippe.Jorrand, Marie.Lalire}@imag.fr

Abstract. Research in quantum computation is looking for the consequences of having information encoding, processing and communication exploit the laws of quantum physics, i.e. the laws of the ultimate knowledge that we have, today, of the foreign world of elementary particles, as described by quantum mechanics. After an introduction to the principles of quantum information processing and a brief survey of the major breakthroughs brought by the first ten years of research in this domain, this paper concentrates on a typically "computer science" way to reach a deeper understanding of what it means to compute with quantum resources, namely on the design of programming languages for quantum algorithms and protocols, and on the questions raised by the semantics of such languages. Special attention is devoted to the process algebraic approach to such languages, through a presentation of QPAlg, the Quantum Process Algebra which is being designed by the authors.

1 From Quantum Physics to Computation

Information is physical: the laws which govern its encoding, processing and communication are bound by those of its unavoidably physical incarnation. In today's informatics, information obeys the laws of Newton's and Maxwell's classical physics: this statement holds all the way from commercial computers down to (up to?) Turing machines and lambda-calculus. Today's computation is classical.

Quantum information processing and communication was born some ten years ago, as a child of two major scientific achievements of the 20th century, namely quantum physics and information sciences. The driving force of research in quantum computation is that of looking for the consequences of having information encoding, processing and communication based upon the laws of quantum physics, i.e. the ultimate knowledge that we have, today, of the foreign world of elementary particles, as described by quantum mechanics. The principles of quantum information processing are very briefly introduced in this section. For a more detailed, but still concise and gentle introduction, see [24]. A pedagogical and rather thorough textbook on quantum computing is [21]. For a dense and theoretically profound presentation, the reader is referred to [16].

J.-P. Banâtre et al. (Eds.): UPP 2004, LNCS 3566, pp. 1 – 16, 2005.

1.1 Four Postulates for Computing

Quantum mechanics, which is the mathematical formulation of the laws of quantum physics, relies on four postulates: (*i*) the state of a quantum system (i.e. a particle, or a collection of particles) is a unit element of a Hilbert space, that is a vector of norm 1 in a *d*-dimensional complex vector space; (*ii*) the evolution of the state of a closed quantum system (i.e. not interacting with its -classical- environment) is deterministic, linear, reversible and characterized by a unitary operator, that is by a *dxd* unitary matrix applied to the state vector; (*iii*) the measurement of a quantum system (i.e. the observation of a quantum system by its -classical- environment) irreversibly modifies the state of the system by performing a projection of the state vector onto a probabilistically chosen subspace of the Hilbert space, with renormalization of the resulting vector, and returns a value (e.g. an integer) to the classical world, which just tells which subspace was chosen; and (*iv*) the state space of a quantum system composed of several quantum subsystems is the tensor product of the state spaces of its components (given two vector spaces P and Q of dimensions p and q respectively, their tensor product is a vector space of dimension pxq).

The question is then: how to take advantage of these postulates to the benefits of computation? The most widely developed approach to quantum computation exploits all four postulates in a rather straightforward manner. The elementary physical carrier of information is a qubit (quantum bit), i.e. a quantum system (electron, photon, ion, ...) with a 2-dimensional state space (postulate *i*); the state of a n-qubit register lives in a 2^n-dimensional Hilbert space, the tensor product of n 2-dimensional Hilbert spaces (postulate *iv*). Then, by imitating in the quantum world the most traditional organization of classical computation, quantum computations are considered as comprising three steps in sequence: first, preparation of the initial state of a quantum register (postulate *iii* can be used for that, possibly with postulate *ii*); second, computation, by means of deterministic unitary transformations of the register state (postulate *ii*); and third, output of a result by probabilistic measurement of all or part of the register (postulate *iii*).

1.2 Quantum Ingredients for Information Processing

These postulates and their consequences can be interpreted from a more informational and computational point of view, thus providing the elementary quantum ingredients which are at the basis of quantum algorithm design:

Superposition. At any given moment, the state of quantum register of n qubits is a vector in a 2^n-dimensional complex vector space, i.e. a vector with at most 2^n non zero complex components, one for each of the 2^n different values on n bits: the basis of this vector space comprises the 2^n vectors $|i\rangle$, for i in $\{0,1\}^n$ ($|i\rangle$ is Dirac's notation for vectors denoting quantum states). This fact is exploited computationally by considering that this register can actually contain a superposition of all the 2^n different values on n bits, whereas a classical register of n bits may contain only one of these values at any given moment.

Quantum Parallelism and Deterministic Computation. Let f be a function from $\{0,1\}^n$ to $\{0,1\}^m$ and x be a quantum register of n qubits initialized in a superposition of all values in $\{0,1\}^n$ (this initialization can be done by one very simple step). Then, computing $f(x)$ is achieved by a deterministic, linear and unitary operation on the state of x: because of linearity, a single application of this operation produces all 2^n values of f in a single computation step. Performing this operation for any, possibly non linear f while obeying the linearity and unitarity laws of the quantum world, requires a register of $n+m$ qubits formed of the register x, augmented with a register y of m qubits. Initialy, y is in any arbitray state $|s\rangle$ on m qubits: before the computation of f, the larger register of $n+m$ qubits contains a superposition of all pairs $|i,s\rangle$ for i in $\{0,1\}^n$. After the computation of f, it contains a superposition of all pairs $|i, s\oplus f(i)\rangle$ for i in $\{0,1\}^n$, where $\oplus$ is bitwise addition modulo 2. It is easy to verify that, for any f, this operation on a register of $n+m$ qubits is unitary (it is its own inverse). In many cases, it will be applied with $s=0$, which results in a superposition of all simpler pairs $|i, f(i)\rangle$ for i in $\{0,1\}^n$.

Probabilistic Measurement and Output of a Result. After f has been computed, all its values $f(i)$, for i in $\{0,1\}^n$, are superposed in the y part (m qubits) of the register of $n+m$ qubits, each of these values facing (in the pair $|i,f(i)\rangle$) their corresponding i which is still stored in the unchanged superposition contained in the x part (n qubits) of that register. Observing the contents of y will return only one value, j, among the possible values of f. This value is chosen with a probability which depends on f since, e.g. if $f(i)=j$ for more than one values of i, the probability of obtaining j as a result will be higher than that of obtaining k if $f(i)=k$ for only one value of I (and the probability of obtaining l if there is no i such that $f(i)=l$ will of course be 0). This measurement also causes the superposition in y to be projected onto the 1-dimensional subspace corresponding to the basis state $|j\rangle$, i.e. the state of the y part collapses to $|j\rangle$, which implies that all other values of f which were previously superposed in y are irreversibly lost.

Interference. Using appropriate unitary operations, the results of the 2^n parallel computations of f over its domain of definition can be made to interfere with each other. Substractive interference will lower the probability of observing some of these value in y, whereas additive interference will increase the probability of observing other values and bring it closer to 1. Because of probabilistic measurement, a major aim of the organization and principles of quantum algorithms will be to assemble the unitary operations for a given computation in such a way that, when a final measurement is applied, a relevant result has a high probability to be obtained.

Entangled States. Measuring y after the computation of f is in fact measuring only m qubits (the y part) among the $n+m$ qubits of a register. The state of this larger register is a superposition of all pairs $|i,f(i)\rangle$ for i in $\{0,1\}^n$ (e.g., in this superposition, there is no pair like $|2,f(3)\rangle$): this superposition is not a free cross-product of the domain $\{0,1\}^n$ of f by its image in $\{0,1\}^m$, i.e. there is a strong correlation between the contents of the x and y parts of the register. As a consequence, if measuring the y part returns a value j, with the state of that part collapsing to the basis state $|j\rangle$, the state of the larger register will itself collapse to a superposition of all remaining pairs $|i,j\rangle$

such that $f(i)=j$. This means that, in addition to producing a value j, the measurement of the y part also causes the state of the x part to collapse to a superposition of all elements of the $f^{-1}(j)$ subset of the domain of f. This correlation between the x and y parts of the register is called entanglement: in quantum physics, the state of a system composed of n sub-systems is not, in general, simply reducible to an n-tuple of the states of the components of that system. Entanglement has no equivalent in classical physics and it constitutes the most powerful resource for quantum information processing and communication.

No-Cloning. A direct consequence of the linearity of all operations that can be applied to quantum states (a two line trivial proof shows it) is that the state of a qubit a (this state is in general an arbitrary superposition, i.e. a vector made of a linear combination of the two basis state vectors |0> and |1>), cannot be duplicated and made the state of another qubit b, unless the state of a is simply either |0> or |1> (i.e. not an arbitrary superposition). This is true of the state of all quantum systems, including of course registers of n qubits used during a quantum computation. In programming terms, this means that the "value" (the state) of a quantum variable cannot be copied into another quantum variable.

These basic quantum ingredients and their peculiarities will of course have far reaching consequences, as soon as algorithm, programming languages and semantic frameworks incorporate and make use of quantum resources.

2 Quantum Algorithms

Richard Feynman launched in 1982 [10] the idea that computation based upon quantum physics would be exponentially more efficient than based upon classical physics. Then, after the pioneering insight of David Deutsch in the mid eighties [8], who showed, by means of a quantum Turing machine, that quantum computing could indeed not, in general, be simulated in polynomial time by classical computing, it was ten years before the potential power of quantum computing was demonstrated on actual computational problems.

2.1 Major Breakthroughs: Quantum Speedups and Teleportation

The first major breakthrough was by Peter Shor [27]: in 1994, he published a quantum algorithm operating in polynomial time ($O(\log^3 N)$) for factoring an integer N, whereas the best classical algorithm is exponential. Two years later, Lov Grover [13] published a quantum algorithm for searching an unordered database of size N, which achieves a quadratic acceleration (it operates in $O(N^{1/2})$) when compared with classical algorithms for the same problem (in $O(N)$). Shor's algorithm relies on a known reduction of the problem of factoring to that of finding the order of a group, or the period of a function: then, since order finding can be achieved by a Fourier Transform, the key of Shor's algorithm is a Quantum Fourier Transform, which is indeed exponentially more efficient than FFT, thanks to quantum parallelism, entanglement and tensor product. Grover's algorithm relies upon a very subtle use of interference,

now known as amplitude amplification, which performs a stepwise increase of the probability of measuring a relevant item in the database, and which brings this probability very close to one after $N^{1/2}$ steps.

Another major result, by Charles Bennet and others in 1993 [3], was the design of theoretical principles leading to a quantum teleportation protocol, which takes advantage of entanglement and of probabilistic measurement: the state of a quantum system a (e.g. a qubit) localized at A's place can be assigned, after having been measured, thus destroyed, to another quantum system b (e.g. another qubit), localized at B's place, without the state of a being known neither by A nor by B, and without neither a, b nor any other quantum system being moved along a trajectory between A and B. It is important to notice that this is not in contradiction with no-cloning: there is still only one instance of the teleported state, whereas cloning would mean that there coexist one original and one copy.

Since then, these results have been generalized and extended to related classes of problems. Shor's algorithm solves an instance of the hidden subgroup problem [19] for abelian groups and a few extensions to non-abelian cases have been designed. In addition to Fourier Transform, order finding and amplitude amplification, other candidates to the status of higher level building blocks for quantum algorithmics have emerged, such as quantum random walks on graphs [15]. Principles for distributed quantum computing have also been studied and successfully applied to a few classes of problems. Very recently, on the basis of amplitude amplification, quadratic and other quantum speedups have been found for several problems on graphs, such as connectivity, minimum spanning tree and single source shortest paths [9].

Teleportation also has been generalized. The measurement used in its original formulation was such that the state eventually obtained for b was the same as the state initially held by a (up to a correcting operation which still had to be applied, depending on the probabilistic outcome of that measurement). By changing the way the measurement is done (in fact, by appropriately rotating the basis upon which the measurement of a will project the state of a), it has been found that the state teleported to b could be not the state initially held by a, but that state to which a rotation, i.e. a unitary operation has been applied. In other words, entanglement and measurement, i.e. the resources needed by teleportation, can be used to simulate computations by unitary tranformations. This has given rise to a whole new direction of research in quantum computation, namely measurement-based quantum computation [14,18,23].

2.2 No Quantum Computation Without Classical Control

There is an implicit, but obvious and ever present invariant in all these different ways of organizing quantum computations and quantum algorithms. Quantum computations operate in the quantum world, which is a foreign and unknowable world. No one in the classical world will ever know what the superposition state of an arbitrary qubit is, the only information one can get is 0 or 1, through measurement, i.e. the classical outcome of a probabilistic projection of the qubit state vector onto |0> or |1>: if one gets |0>, the only actual information which is provided about the state before measurement is that it was not |1>, because |0> and |1> are orthogonal vectors. Then, for