Germán Puebla (Ed.)

Logic-Based Program Synthesis and Transformation

LNCS 4407

16th International Symposium, LOPSTR 2006
Venice, Italy, July 2006
Revised Selected Papers

Springer

Germán Puebla (Ed.)

Logic-Based Program Synthesis and Transformation

16th International Symposium, LOPSTR 2006
Venice, Italy, July 12-14, 2006
Revised Selected Papers

Springer

Volume Editor

Germán Puebla
Technical University of Madrid (UPM), School of Computer Science
Campus de Montegancedo, 28660 Boadilla del Monte (Madrid), Spain
E-mail: german@fi.upm.es

Library of Congress Control Number: 2007922566

CR Subject Classification (1998): F.3.1, D.1.1, D.1.6, D.2.4, I.2.2, F.4.1

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

ISSN 0302-9743
ISBN-10 3-540-71409-X Springer Berlin Heidelberg New York
ISBN-13 978-3-540-71409-5 Springer Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12035236 06/3142 5 4 3 2 1 0

Lecture Notes in Computer Science 4407

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

Lecture Notes in Computer Science

For information about Vols. 1–4316

please contact your bookseller or Springer

Vol. 4429: R. Lu, J.H. Siekmann, C. Ullrich (Eds.), Cognitive Systems. X, 161 pages. 2007. (Sublibrary LNAI).

Vol. 4424: O. Grumberg, M. Huth (Eds.), Tools and Algorithms for the Construction and Analysis of Systems. XX, 738 pages. 2007.

Vol. 4423: H. Seidl (Ed.), Foundations of Software Science and Computational Structures. XVI, 379 pages. 2007.

Vol. 4422: M.B. Dwyer, A. Lopes (Eds.), Fundamental Approaches to Software Engineering. XV, 440 pages. 2007.

Vol. 4421: R. De Nicola (Ed.), Programming Languages and Systems. XVII, 538 pages. 2007.

Vol. 4420: S. Krishnamurthi, M. Odersky (Eds.), Compiler Construction. XIV, 233 pages. 2007.

Vol. 4415: P. Lukowicz, L. Thiele, G. Tröster (Eds.), Architecture of Computing Systems - ARCS 2007. X, 297 pages. 2007.

Vol. 4414: S. Hochreiter, R. Wagner (Eds.), Bioinformatics Research and Development. XVI, 482 pages. 2007. (Sublibrary LNBI).

Vol. 4407: G. Puebla (Ed.), Logic-Based Program Synthesis and Transformation. VIII, 237 pages. 2007.

Vol. 4405: L. Padgham, F. Zambonelli (Eds.), Agent-Oriented Software Engineering VII. XII, 225 pages. 2007.

Vol. 4403: S. Obayashi, K. Deb, C. Poloni, T. Hiroyasu, T. Murata (Eds.), Evolutionary Multi-Criterion Optimization. XIX, 954 pages. 2007.

Vol. 4399: X. Llorà, T. Kovacs, K. Takadama, P.L. Lanzi, S.W. Wilson, W. Stolzmann (Eds.), Learning Classifier Systems. XII, 345 pages. 2007. (Sublibrary LNAI).

Vol. 4397: C. Stephanidis, M. Pieper (Eds.), Universal Access in Ambient Intelligence Environments. XV, 467 pages. 2007.

Vol. 4396: J. García-Vidal, L. Cerdà-Alabern (Eds.), Wireless Systems and Mobility in Next Generation Internet. IX, 271 pages. 2007.

Vol. 4394: A. Gelbukh (Ed.), Computational Linguistics and Intelligent Text Processing. XVI, 648 pages. 2007.

Vol. 4393: W. Thomas, P. Weil (Eds.), STACS 2007. XVIII, 708 pages. 2007.

Vol. 4392: S.P. Vadhan (Ed.), Theory of Cryptography. XI, 595 pages. 2007.

Vol. 4390: S.O. Kuznetsov, S. Schmidt (Eds.), Formal Concept Analysis. X, 329 pages. 2007. (Sublibrary LNAI).

Vol. 4389: D. Weyns, H.V.D. Parunak, F. Michel (Eds.), Environments for Multi-Agent Systems III. X, 273 pages. 2007. (Sublibrary LNAI).

Vol. 4385: K. Coninx, K. Luyten, K.A. Schneider (Eds.), Task Models and Diagrams for Users Interface Design. XI, 355 pages. 2007.

Vol. 4384: T. Washio, K. Satoh, H. Takeda, A. Inokuchi (Eds.), New Frontiers in Artificial Intelligence. IX, 401 pages. 2007. (Sublibrary LNAI).

Vol. 4383: E. Bin, A. Ziv, S. Ur (Eds.), Hardware and Software, Verification and Testing. XII, 235 pages. 2007.

Vol. 4381: J. Akiyama, W.Y.C. Chen, M. Kano, X. Li, Q. Yu (Eds.), Discrete Geometry, Combinatorics and Graph Theory. XI, 289 pages. 2007.

Vol. 4380: S. Spaccapietra, P. Atzeni, F. Fages, M.-S. Hacid, M. Kifer, J. Mylopoulos, B. Pernici, P. Shvaiko, J. Trujillo, I. Zaihrayeu (Eds.), Journal on Data Semantics VIII. XV, 219 pages. 2007.

Vol. 4378: I. Virbitskaite, A. Voronkov (Eds.), Perspectives of Systems Informatics. XIV, 496 pages. 2007.

Vol. 4377: M. Abe (Ed.), Topics in Cryptology – CT-RSA 2007. XI, 403 pages. 2006.

Vol. 4376: E. Frachtenberg, U. Schwiegelshohn (Eds.), Job Scheduling Strategies for Parallel Processing. VII, 257 pages. 2007.

Vol. 4374: J.F. Peters, A. Skowron, I. Düntsch, J. Grzymała-Busse, E. Orłowska, L. Polkowski (Eds.), Transactions on Rough Sets VI, Part I. XII, 499 pages. 2007.

Vol. 4373: K. Langendoen, T. Voigt (Eds.), Wireless Sensor Networks. XIII, 358 pages. 2007.

Vol. 4372: M. Kaufmann, D. Wagner (Eds.), Graph Drawing. XIV, 454 pages. 2007.

Vol. 4371: K. Inoue, K. Satoh, F. Toni (Eds.), Computational Logic in Multi-Agent Systems. X, 315 pages. 2007. (Sublibrary LNAI).

Vol. 4370: P.P Lévy, B. Le Grand, F. Poulet, M. Soto, L. Darago, L. Toubiana, J.-F. Vibert (Eds.), Pixelization Paradigm. XV, 279 pages. 2007.

Vol. 4369: M. Umeda, A. Wolf, O. Bartenstein, U. Geske, D. Seipel, O. Takata (Eds.), Declarative Programming for Knowledge Management. X, 229 pages. 2006. (Sublibrary LNAI).

Vol. 4368: T. Erlebach, C. Kaklamanis (Eds.), Approximation and Online Algorithms. X, 345 pages. 2007.

Vol. 4367: K. De Bosschere, D. Kaeli, P. Stenström, D. Whalley, T. Ungerer (Eds.), High Performance Embedded Architectures and Compilers. XI, 307 pages. 2007.

Vol. 4366: K. Tuyls, R. Westra, Y. Saeys, A. Nowé (Eds.), Knowledge Discovery and Emergent Complexity in Bioinformatics. IX, 183 pages. 2007. (Sublibrary LNBI).

Vol. 4364: T. Kühne (Ed.), Models in Software Engineering. XI, 332 pages. 2007.

Vol. 4362: J. van Leeuwen, G.F. Italiano, W. van der Hoek, C. Meinel, H. Sack, F. Plášil (Eds.), SOFSEM 2007: Theory and Practice of Computer Science. XXI, 937 pages. 2007.

Vol. 4361: H.J. Hoogeboom, G. Păun, G. Rozenberg, A. Salomaa (Eds.), Membrane Computing. IX, 555 pages. 2006.

Vol. 4360: W. Dubitzky, A. Schuster, P.M.A. Sloot, M. Schroeder, M. Romberg (Eds.), Distributed, High-Performance and Grid Computing in Computational Biology. X, 192 pages. 2007. (Sublibrary LNBI).

Vol. 4358: R. Vidal, A. Heyden, Y. Ma (Eds.), Dynamical Vision. IX, 329 pages. 2007.

Vol. 4357: L. Buttyán, V. Gligor, D. Westhoff (Eds.), Security and Privacy in Ad-Hoc and Sensor Networks. X, 193 pages. 2006.

Vol. 4355: J. Julliand, O. Kouchnarenko (Eds.), B 2007: Formal Specification and Development in B. XIII, 293 pages. 2006.

Vol. 4354: M. Hanus (Ed.), Practical Aspects of Declarative Languages. X, 335 pages. 2006.

Vol. 4353: T. Schwentick, D. Suciu (Eds.), Database Theory – ICDT 2007. XI, 419 pages. 2006.

Vol. 4352: T.-J. Cham, J. Cai, C. Dorai, D. Rajan, T.-S. Chua, L.-T. Chia (Eds.), Advances in Multimedia Modeling, Part II. XVIII, 743 pages. 2006.

Vol. 4351: T.-J. Cham, J. Cai, C. Dorai, D. Rajan, T.-S. Chua, L.-T. Chia (Eds.), Advances in Multimedia Modeling, Part I. XIX, 797 pages. 2006.

Vol. 4349: B. Cook, A. Podelski (Eds.), Verification, Model Checking, and Abstract Interpretation. XI, 395 pages. 2007.

Vol. 4348: S.T. Taft, R.A. Duff, R.L. Brukardt, E. Ploedereder, P. Leroy (Eds.), Ada 2005 Reference Manual. XXII, 765 pages. 2006.

Vol. 4347: J. Lopez (Ed.), Critical Information Infrastructures Security. X, 286 pages. 2006.

Vol. 4346: L. Brim, B. Haverkort, M. Leucker, J. van de Pol (Eds.), Formal Methods: Applications and Technology. X, 363 pages. 2007.

Vol. 4345: N. Maglaveras, I. Chouvarda, V. Koutkias, R. Brause (Eds.), Biological and Medical Data Analysis. XIII, 496 pages. 2006. (Sublibrary LNBI).

Vol. 4344: V. Gruhn, F. Oquendo (Eds.), Software Architecture. X, 245 pages. 2006.

Vol. 4342: H. de Swart, E. Orłowska, G. Schmidt, M. Roubens (Eds.), Theory and Applications of Relational Structures as Knowledge Instruments II. X, 373 pages. 2006. (Sublibrary LNAI).

Vol. 4341: P.Q. Nguyen (Ed.), Progress in Cryptology - VIETCRYPT 2006. XI, 385 pages. 2006.

Vol. 4340: R. Prodan, T. Fahringer, Grid Computing. XXIII, 317 pages. 2007.

Vol. 4339: E. Ayguadé, G. Baumgartner, J. Ramanujam, P. Sadayappan (Eds.), Languages and Compilers for Parallel Computing. XI, 476 pages. 2006.

Vol. 4338: P. Kalra, S. Peleg (Eds.), Computer Vision, Graphics and Image Processing. XV, 965 pages. 2006.

Vol. 4337: S. Arun-Kumar, N. Garg (Eds.), FSTTCS 2006: Foundations of Software Technology and Theoretical Computer Science. XIII, 430 pages. 2006.

Vol. 4336: V.R. Basili, H.D. Rombach, K. Schneider, B. Kitchenham, D. Pfahl, R.W. Selby, Empirical Software Engineering Issues. XVII, 194 pages. 2007.

Vol. 4335: S.A. Brueckner, S. Hassas, M. Jelasity, D. Yamins (Eds.), Engineering Self-Organising Systems. XII, 212 pages. 2007. (Sublibrary LNAI).

Vol. 4334: B. Beckert, R. Hähnle, P.H. Schmitt (Eds.), Verification of Object-Oriented Software. XXIX, 658 pages. 2007. (Sublibrary LNAI).

Vol. 4333: U. Reimer, D. Karagiannis (Eds.), Practical Aspects of Knowledge Management. XII, 338 pages. 2006. (Sublibrary LNAI).

Vol. 4332: A. Bagchi, V. Atluri (Eds.), Information Systems Security. XV, 382 pages. 2006.

Vol. 4331: G. Min, B. Di Martino, L.T. Yang, M. Guo, G. Ruenger (Eds.), Frontiers of High Performance Computing and Networking – ISPA 2006 Workshops. XXXVII, 1141 pages. 2006.

Vol. 4330: M. Guo, L.T. Yang, B. Di Martino, H.P. Zima, J. Dongarra, F. Tang (Eds.), Parallel and Distributed Processing and Applications. XVIII, 953 pages. 2006.

Vol. 4329: R. Barua, T. Lange (Eds.), Progress in Cryptology - INDOCRYPT 2006. X, 454 pages. 2006.

Vol. 4328: D. Penkler, M. Reitenspiess, F. Tam (Eds.), Service Availability. X, 289 pages. 2006.

Vol. 4327: M. Baldoni, U. Endriss (Eds.), Declarative Agent Languages and Technologies IV. VIII, 257 pages. 2006. (Sublibrary LNAI).

Vol. 4326: S. Göbel, R. Malkewitz, I. Iurgel (Eds.), Technologies for Interactive Digital Storytelling and Entertainment. X, 384 pages. 2006.

Vol. 4325: J. Cao, I. Stojmenovic, X. Jia, S.K. Das (Eds.), Mobile Ad-hoc and Sensor Networks. XIX, 887 pages. 2006.

Vol. 4323: G. Doherty, A. Blandford (Eds.), Interactive Systems. XI, 269 pages. 2007.

Vol. 4322: F. Kordon, J. Sztipanovits (Eds.), Reliable Systems on Unreliable Networked Platforms. XIV, 317 pages. 2007.

Vol. 4320: R. Gotzhein, R. Reed (Eds.), System Analysis and Modeling: Language Profiles. X, 229 pages. 2006.

Vol. 4319: L.-W. Chang, W.-N. Lie (Eds.), Advances in Image and Video Technology. XXVI, 1347 pages. 2006.

Vol. 4318: H. Lipmaa, M. Yung, D. Lin (Eds.), Information Security and Cryptology. XI, 305 pages. 2006.

Vol. 4317: S.K. Madria, K.T. Claypool, R. Kannan, P. Uppuluri, M.M. Gore (Eds.), Distributed Computing and Internet Technology. XIX, 466 pages. 2006.

¥452.00元

Preface

This volume contains a selection of papers presented at LOPSTR 2006, the 16th International Symposium on Logic-Based Program Synthesis and Transformation, held in Venice, Italy, July, 12-14 2006.

The aim of the LOPSTR series is to stimulate and promote international research and collaboration on logic-based program development. Previous LOPSTR events were held in London (2005, 2000), Verona (2004), Uppsala (2003), Madrid (2002), Paphos (2001), Venice (1999), Manchester (1998, 1992, 1991), Leuven (1997), Stockholm (1996), Arnhem (1995), Pisa (1994), and Louvain-la-Neuve (1993).

We would like to thank all those who submitted contributions to LOPSTR. Overall, we received 41 submissions (29 full papers and 12 extended abstracts). Each submission received at least three reviews. The committee decided to accept nine of these full papers for presentation and for inclusion in the final conference proceedings. In addition, eight extended abstracts, including two tool demonstrations, were accepted for presentation only. After the conference, authors of extended abstracts describing research judged to be mature enough for possible publication in the present volume were invited to submit full papers. In this second reviewing process, five additional papers were accepted for publication in the current LNCS volume, together with revised versions of the nine full papers previously accepted.

We would also like to thank Shaz Qadeer and Massimo Marchiori for agreeing to give invited talks and for their contribution to these proceedings.

I am very grateful to the authors of the papers, the reviewers, and in particular to the members of the Program Committee for their invaluable help. Thanks also to Andrei Voronkov for his support with the use of EasyChair, which greatly simplified the submission, reviewing and discussion process, as well as the preparation of the proceedings.

LOPSTR 2006 was co-located with PPDP 2006 (ACM Symposium on Principles and Practice of Declarative Programming) and ICALP 2006 (International Colloquium on Automata, Languages and Programming).

My warmest thanks go to Sabina Rossi (Local Arrangements Chair), who was always willing to help in any aspect of the organization of the event. Special thanks also to Annalisa Bossi and Michele Bugliesi who, together with Sabina, took care of the overall planning and local organization of LOPSTR 2006.

December 2006

Germán Puebla

Conference Organization

Program Chair

Germán Puebla

Program Committee

Slim Abdennadher
Roberto Bagnara
Gilles Barthe
John Gallagher
Robert Glück
Michael Hanus
Patricia M. Hill
Kazuhiko Kakehi
Andy King
Michael Leuschel
Fred Mesnard
Sabina Rossi
Grigore Rosu
Wim Vanhoof
Germán Vidal

Local Organization

Sabina Rossi (Local Arrangements Chair)
Annalisa Bossi
Michele Bugliesi

External Reviewers

James Avery, Bernd Braßel, Diego Calvanese, Stephen Bond, Alvaro Cortes, Guillaume Dufay, Santiago Escobar, Marc Fontaine, Samir Genaim, Mark Hills, Frank Huch, Dongxi Liu, Wafik Boulos Lotfallah, Thomas Lukasiewicz, Damiano Macedonio, Claude Marché, Viviana Mascardi, Thierry Massart, Kazutaka Matsuda, Nancy Mazur, Antoine Miné, Torben Mogensen, Akimasa Morihata, Klaus Ostermann, Etienne Payet, Andrea Pescetti, Alberto Pettorossi, Carla Piazza, David Pichardie, Andrei Popescu, Maurizio Proietti, Traian Florin Serbanuta, Fausto Spoto, Xavier Urbain, Brent Venable, Tetsuo Yokoyama, Enea Zaffanella.

Table of Contents

Invited Talks

Program Development

Partial Evaluation and Program Transformation

Security and Synthesis

Debugging and Testing

Termination and Analysis

How to Talk to a Human: The Semantic Web and the Clash of the Titans

Massimo Marchiori[1,2]

[1] University of Padua (UNIPD)
massimo@math.unipd.it
[2] Utility Labs (UTILABS)
massimo@utilabs.org

Abstract. The Semantic Web has managed to produce an enormous buzzword. However, despite it cannot be considered a new technology anymore, it didn't fly off yet, and has remained unexpressed in its potentials. In this article we try to analyze the possible reasons, and also the tension that the Semantic Web has with XML. We emphasize the need for consideration of the more comprehensive social environment, together with a more formal modeling of the mechanics of the Web and its information flows.

1 The Semantic Web and XML: The Eternal Quest

The Semantic Web (mostly, in its RDF [1] incarnation) and XML have been often seen as two distinct worlds, and as such, each of them has a community of people who think the other side of the fence is doing things "the wrong way".

Given XML's success, and the current dual lack of success of the Semantic Web/RDF, it is normal that the latter has been often criticized, using the following "fundamental question":

Q: What can you do with RDF that you can't do with XML?

The fundamental question is both tricky and crucial. This question has been source of embarrassment, and of misunderstandings, for both worlds, and has somehow contributed to the lack of proper understanding of the potential of the Semantic Web in the context of the bigger XML world.

We were saying the question is tricky. The classic general answer which is given is:

Q: What can you do with RDF that you can't do with XML?

A: Semantics!

This usually leaves the XML-World unsatisfied, because this is in fact a very fuzzy answer. Saying that with RDF you can do semantics, equals more or less to say that with the Semantic Web you can do... semantics, which doesn't sound too good to critical eyes. So then, the "socratic dialogue" goes on, and the XML-World usually replies with

G. Puebla (Ed.): LOPSTR 2006, LNCS 4407, pp. 1–14, 2007.

XML-World: What do you mean?

More or less, the debate between RDF-World and XML-World then goes on like this:

RDF-World: With RDF I can do X.

XML-World, Well, I can do X with XML too, so what?

RDF-World: But, with RDF I can do Y.

XML-World, Well, I can do Y with XML as well, so what?

(and so on, and so on...)

The point is that the answer is in fact quite easy, and it is one that few people in RDF-World would dare to mention explicitly:

Q: What can you do with RDF that you can't do with XML?

A: Nothing!

This comes trivially from the fact that RDF is XML, and therefore, there's no magic in RDF: RDF is just a dialect of XML, and as such, there's nothing RDF can do "more" than XML: the question, posed this way, is just bogus.

But so, does this mean the XML world is right, and that the Semantic Web is superfluous?

2 The Semantic Web to the Rescue: Closed vs. Open Worlds

The answer to the previous question is not that easy: it really depends on what level of precision we want to analyze. It is certainly true that with XML you can do anything you want, but that doesn't prevent RDF (and the related tower of technologies) to be a successful dialect/specialization of XML, like there are many around. But specialization for what, precisely?

XML has been labeled as the best invention after peanut butter: versatile, flexible, powerful. However, there is one thing for which XML, at least apparently, doesn't work so well: aggregating information.

XML's strength is its specialization capabilities: given an information locale, everybody can easily write a local dialect to express that information. In other words, XML works extremely well in the *closed world* context: an environment where there is a centralized vocabulary control. However, there is another scenario, which didn't fit the original design of XML: the *open-world model*, where there is no centralized vocabulary control. In such scenarios, everybody can develop its own local dialect, and then the big problem is how to exchange information between the different vocabularies, integrating various information sources that have no control over each other. Like for the Tower of Babel, where

the multitude of languages has been the disgrace of Humanity, in the open-world model the different languages can provoke heavy interoperability problems (what linguistics call very appropriately the Lost in translations effect).

RDF, more or less consciously, was designed with this fundamental goal in mind (besides the related "give more semantics" mantra): reducing almost to zero the complexity of aggregating information (which, essentially, becomes a merge of graphs). The connections among information pieces are established via the URIs: so, when merging graphs, nodes are considered equal if they have the same URI. Therefore, URIs become the fundamental key to distinguish web object. This choice is compatible, and actually stems from, one of the very first Web Axioms stated by Tim Berners-Lee (the so-called Universality 2 axiom, cf. [2] and compare with the later [3]): meaningful resources on the Web should be identified by URIs.

Thus, RDF is (also) XML, but RDF has been designed to work in the open-world model: while XML works better in the closed-world model, RDF does in the open-world model.

3 Just Aggregation?

So, a first important point that distinguishes "generic" XML from RDF is the complexity of information aggregation. While being an important point, that alone doesn't give the whole picture.

In fact, the Web is, as a whole, semantically speaking, a huge open-world model: so, how come that the Semantic Web didn't rapidly gain success? Something must have gone wrong, and to trace that, we need to start back from the original definition that Tim Berners-Lee gave of the Semantic Web: an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation". Computers... and people! What about the people?

4 The Benefits

What is missing in the equation is the *utilization model*, i.e., the complete benefits (goals) that the new technology is supposed to provide.

Saying that RDF "works better" in the open-world model is a simplistic assertion, as we haven't quite defined what "better" means. If better means aggregating information, the assertion is correct. But aggregation alone isn't what the Semantic Web promise to do (if it were so, the benefits alone wouldn't be quite clear): the goals of the Semantic Web are more ambitious, and for that reason, the original idea of the Semantic Web includes the well-known "Semantic Web tower" (see for example [4]), i.e., a full tower of technologies that better describe the operational model, and therefore help clarifying the benefits.

So for instance, aggregation of information isn't much helpful if we don't have a clear working model that allows us to benefit from that feature. In order to exploit information aggregation we can then for example also include a logic into

the picture: a logic allows to make deductions, and so in principle augments by far the benefits of having aggregated information on the Web. Initial step into this direction have been done with the RDF Semantics, RDF-Schema, OWL, and this line has been continuing more recently with the work of W3C's RIF effort, devoted to specify a Rules Interchange Format that will allow even more flexibility in "programming" the rules shaping web information. This is all consistent with the big view of the Semantic Web Tower.

5 The Costs

But then, there is also the other side of the coin, the dual part that has to be considered every time that we want to analyze the behavior of a successful technology: the *cost factor*.

The overall cost is in general a complex thing to compute, but roughly, it can be seen as the sum of two components: the *technological cost* (the cost for the machine), and the *social cost* (the cost for the people). We can summarize the concept this way:

$$\text{Cost} = \text{Technology} + \text{Society}$$

Both aspects, technology and society, are equally very important. What have happened so far is that the societal cost of the Semantic Web hasn't been object of much attention, and the whole design has been centered on the technological cost, making best efforts to ensure that the technologies in the Semantic Web Tower would have a relatively low technological cost. But in the overall Semantic Web operational model, the scenario is much bigger than just the computational complexity of a logic: it includes the much wider scenario of the Web, its information flows, machines, and the people. Therefore, we need to rethink the situation and not just wear the eyeglasses of the technologist, caring mostly about the computers (classic semantic web stack). Sure, there is the need to monitor and balance the technical cost, but also to consider at least another dimension for the social cost (what has been called the *P axis*, P as Perception/People, in [5]).

Only when we have a complete measure for the cost we can proceed to measure the cost/benefit ratio (shortly, C/B), which is a major indicator of success, especially in environment like the Web.

5.1 The C/B Ratio

The C/B ratio provides a uni-dimensional space that can give a rough estimate of the chances of success of a technology (ranging from 0, the optimum, to infinity, the worst). Minimizing the cost/benefit can happen in a variety of ways, depending on the balance between C and B. In the Web, the important thing to take into account is the dynamics of C and B within the web environment and the users. For instance, in a web-wide application cost usually grows at least linearly with the size of the web (or of the sub-web/community taken into consideration), which can be extremely dangerous. On the other hand, B also in

such applications usually depends on the size of the user base, which is very low in the starting adoption phase. Therefore, if we are not careful the corresponding dynamic system will not lead to a success situation, because the too high C/B in the initial phases will prevent an evolution that makes the C/B decrease and reach a wide enough user base. So, in order to produce a network effect, either the initial cost has to be extremely low, or the initial benefit has to be very high.

5.2 The Cost of URIs

The previous C/B discussion then naturally leads to consider: what are the costs of the Semantic Web? An interesting exercise is to measure the technological cost for the semantic web architecture (e.g. in the Semantic Web tower). The analysis will then reveal, in fact, a nice result: the technological (computational) cost is usually low/moderate according to where one sits in the Tower (although interestingly, even in this respect, computational cost has started to grow a lot, see for instance the logic behind the higher layer of OWL). However, when one views at the historical progression of the Semantic Web (still ongoing...), the situation is that there is an overgrowing set of specifications: RDF Model & Syntax / RDF Schema / RDF/XML Syntax revised / RDF Vocabulary Description Language / RDF Concepts and Abstract Syntax / RDF Semantics / OWL (OWL-DL / OWL-Full)/ SPARQL / RDF-A / Rules... and the list is still growing.

So, what has been happening here? Will the user be able to sustain the social complexity that these layers are going to produce? The answer, for the moment, is in front of everybody: not yet. The overall cost seems too high for the moment. And this comes from a variety of factor, given that as said, the scenario to consider is much bigger than what has been formally analyzed so far (computational complexity of logics): the Web, the people, information flows.

For instance, let's just revisit the basic association mechanism of the semantic web: aggregation via URIs. An old gag that used to be around in the semantic web circles was the following:

Q: How many Semantic Web scientists does it take to change a light bulb?
A: Ten. One to screw the bulb, and nine to agree on what a light bulb is.

This gag is significant for the suggestion it is giving: it's hard to all agree on a concept. If URIs are meant to be identification names, they are the centralized part of an otherwise decentralized and distribute environment, the web. But how to achieve consensus without control? In other words, there is a significant *social problem* with URIs when they are used as universal aggregators of information. This gives raise to the *URI Variant problem*: in general, there can be many variants (URI) for the same concept. The URI Variant problem is particularly bad in view of the *URI Variant Law*: utility of a URI can decrease exponentially with the number of its variants (in other words, the worst-case is exponential).

This is not enough, because the social problem is not just on what common URI to agree, among many. There is also the other side of the coin, which is much more difficult: how to agree on the semantics of a specific URI. This is sometimes called the *URI meaning problem*: in other words, for two different

people in the web the same URI might well mean different things (after all, there is no centralized interpretation for URIs). This problem is rather severe, because it does not simply affect computational complexity (like the URI variant), but deeply touch the relationship between the web and the people who interact with/in it.

So, all in all, what seemed a strength of the Semantic Web, i.e., almost zero cost for aggregating information, is now revealing deeper faces: while the direct technological cost is indeed very low, there is an underlying social cost that is in fact quite high.

Therefore, this extra variable of the social cost, makes the original simplistic observation, that information aggregation in the Semantic Web is very easy and effective, not quite true any more, and emphasizes the lack of a precise operational model and consequent cost/benefit analysis that have occurred so far.

5.3 Another Perspective: Lost in Navigation

Social costs are not limited to URIs, of course, but they can pervade the same data model. Data structures can have a rigid architecture, or lean towards a more liberal framework, therefore going from the areas of structured data, passing thru the intermediate realm of semi-structured data, and ending in the opposite extreme, the area of unstructured data. Within this wide spectrum, we find for instance in small-size data management on one extreme (structured) spreadsheets and the table model, and on the other extreme (unstructured) things like Zig-Zag, the innovative (for the time) concept by Ted Nelson (cf. [6]. In large(r)-size data management, going on with the parallelism, we find relational databases and the relational model (structured), then we can proceed with XML (semi-structured), then ending with RDF (unstructured).

It is therefore interesting to follow the parallelism, and note that the previous unstructured models (like Zig-Zag) didn't have much success, while the more structured ones did. What are the main reasons? This can be explained by using the so-called *Heisenberg Principle for data handling*: If you stretch the flexibility aspect (benefit), you lose in efficiency (cost).

Note that here efficiency doesn't just mean computational efficiency, but efficiency in-the-large, also for the user. In fact, preliminary studies by the author shows that one can quantify the degree of lost in navigation (that is to say, informally, the capability by the user to grasp the data structure, and to navigate without errors in it): the lost in navigation effect increases (not surprisingly) from structured to semi-structured to unstructured. What is more surprising is that there is quite a gap when passing to unstructured models like RDF and graph-like ones. In other words, the amount of flexibility that these kinds of models give, has a very high price that the user needs to pay. This can explain more formally why unstructured data didn't gain so far the wide success they were expected to. What this also means is that, in order to lower the cost/benefit ratio, there is the need for extra efforts to raise the benefit.

On a related side, the gap occurring inbetween structured and semi-structured data is comparatively rather small, which might also explain why technologies like XML managed to gain success, despite the initial dominant position of structured data approaches.

5.4 Technologies Examples: The Good and the Bad

Let's sweep out of the XML / Semantic Web scenario, and for the sake of illustration, try to see some other examples of more specific technologies and their related cost/benefit.

A first pair of examples is interesting, and comes from privacy technologies developed by the author: P3P and APPEL.

P3P (standing for Platform for Privacy Preferences) is the world standard for Privacy on the Web ([7]). Analyzing the P3P specification will easily show that the P3P technical cost is very low (in fact, not surprisingly, as this was a crucial requirement). As far as the Social Cost is concerned, it is moderate for site maintainers: the moderate complexity essentially stems from building the privacy policy for the site, although this can be ameliorated by specific tools, and in any case approximate policies can be written that are much easier; complementary, publishing the privacy policy is very easy and has a very low cost.

Now let's turn our attention to the benefits side. The benefit is moderate for users (this can be evaluated by using the many privacy surveys available), whereas, interestingly enough, it is very high for site maintainers. The reason? When Internet Explorer passed from version 5 to version 6, it actually incorporated the P3P technology, and in a very stringent way: sites not P3P compliant had severe problems and their cookies were essentially blocked by the browsers. This crucial step provoked a huge rise in the benefit of implementing P3P (even if just at the site maintainers side), and therefore boosted the C/B ratio of P3P, despite cost wasn't low (this C/B boost can be also verified by using the statistical P3P dashboards published by Ernst&Young on the subject).

Now, we want also to consider the other side of the coin, as we said initially that we were going to consider a pair of technologies: P3P and APPEL. APPEL [8] is the companion technology to P3P: the acronym stands for A Privacy Preference Language, and it is a language that enables users (via their browser, for instance), to program on a fine level whether or not to enter a web site, according to the privacy level the site itself provides.

The technological cost for APPEL is moderate, as it can be easily seen. On the other hand, the social cost is high: users need to get knowledge of the privacy possibilities, and to adequately shape a set of preferences. This was too much, given both the relative user interest in privacy (versus content, for instance), and the complexity of programming/shaping a fine level behavior. On the other hand, the benefit here was also relatively small, as the additional privacy control wasn't enough more than for instance some easy pre-defined levels (that Internet Explorer in fact implemented). As a result, the C/B ratio never got sufficiently low, and APPEL didn't fly (in fact, it was never promoted to W3C Recommendation, and remains a proposed technology).