Logic at Botik '89

Lecture Notes in Computer Science

Edited by G. Goos and J. Hartmanis

363

A.R. Meyer M.A. Taitslin (Eds.)

Logic at Botik '89

Symposium on Logical Foundations of Computer Science
Pereslavl-Zalessky, USSR, July 3–8, 1989
Proceedings

Springer-Verlag

Berlin Heidelberg New York London Paris Tokyo Hong Kong

Volume Editors

Albert R. Meyer
MIT, Laboratory for Computer Science
545 Technology Square, Cambridge, MA 02139, USA

Michael A. Taitslin
62 Mojaiskogo Street, Apt. 265
Kalinin 170043, USSR

CR Subject Classification (1987): F.3, D.3.1, D.3.3, D.2.4, F.4.1

ISBN 3-540-51237-3 Springer-Verlag Berlin Heidelberg New York
ISBN 0-387-51237-3 Springer-Verlag New York Berlin Heidelberg

Printing and binding: Druckhaus Beltz, Hemsbach/Bergstr.
2145/3140-543210 – Printed on acid-free paper

Andrei Petrovich Ershov

1931-1988

Andrei Petrovich Ershov, Academician of the USSR Academy of Sciences, died on December 8, 1988, after a long illness. He was a prominent Computer Scientist and Mathematician who chaired the Scientific Council on Cybernetics of the USSR Academy of Sciences.

Ershov possessed a profound sense of responsibility for the future of scientific research and was deeply involved in the development of Computer Science in the USSR. His work ranged from the promotion of computer education in the Soviet Union to the organization of researchers in the field of Theoretical Computer Science.

He was the author of fundamental works in the fields of theoretical programming, programming languages theory, theory of program schemata, and mixed computations. One of the first compilers for the ALGOL-60 programming language was developed under his supervision.

Ershov was known and regarded highly throughout the world for his efforts over two decades in promoting international contacts among Computer Scientists. He continued in this role to the end of his career, playing a significant part on the organizing committee of the *Logic at Botik '89* Symposium despite his illness.

Grateful memories remain in the hearts of his colleagues and friends as they grieve for Andrei Petrovich Ershov.

Foreword

Although the Program Systems Institute of the USSR Academy of Sciences has only been located in Pereslavl-Zalessky since 1986, this symposium, *Logic at Botik, '89,* is not our first experience in organizing a meeting in the field of logical foundations of Computer Science. Other such meetings took place here previously within the framework of symposia on Artificial Intelligence; the largest of them was held in October 1986.

The Program Systems Institute pays careful attention to the development of theoretical foundations of Computer Science, in particular, to the investigation of topics in Logic, because we believe that without advanced fundamental research, it is impossible to achieve success in the field of applications, *e.g.,* in the development of software for scientific use. Research in the field of theoretical foundations is, in the final analysis, the only way that promises to advance programming from a craft to an industrial discipline.

I believe that successful development of science is impossible while barriers separate scientists. The Soviet Union is viewed by some as having fallen behind developed countries in the field of computer technology—although in theoretical Computer Science everything is seen to be OK. However, I am convinced that the "Iron Curtain" greatly damaged the development of science throughout the world, and not only in the Soviet Union. The new policy of *glasnost* provides an opportunity to change the situation.

These are the reasons why I championed the proposals of my colleagues to organize the international symposium on logical foundations of Computer Science *"Logic at Botik '89"* in Pereslavl-Zalessky. I hope that such meetings will become a tradition.

The papers in this volume, to my mind, demonstrate that the program is rich in content, successfully combining promising theoretical works and good applications of the theory. I would like to express my acknowledgements to all the members of the international program committee for doing the job of selecting papers and compiling the volume.

The members of the Organizing Committee, *Logic at Botik '89*, were

A. K. Ailamazian, *Chair*
A.P. Ershov
A.J. Kfoury
K.A. Knorre
G.F. Shvarts
E.V. Stas
A.P. Stolboushkin
P. Urzyczyn

Invaluable help and support was given to the symposium by the Vice-President of the USSR Academy of Sciences, Academician Evgeny P. Velikhov. We met with warm support from town authorities of Pereslavl-Zalessky, especially that of Mayor Vladimir I. Shesterniov.

I am most grateful to A.P. Ershov who, despite illness, found the opportunity and strength to participate in the work of the organizing committee before his untimely death in December, 1988. I would like to thank A.P. Stolboushkin who played the largest part in organizational matters.

I express my sincere gratitude to all my colleagues who did their best to bring the symposium into being.

Prof. Alfred K. Ailamazian
Director of Program Systems
Institute of the USSR Academy of Sciences

Pereslavl-Zalessky
February 1989

Preface

This volume is the proceedings of *Logic at Botik, '89*, a symposium on logical foundations of Computer Science organized by the Program Systems Institute of the USSR Academy of Sciences and held at Pereslavl-Zalessky, USSR, July 3–8, 1989

Topics of interest mentioned in the call for papers were:

Complexity of formal systems
Constructive mathematics in computer science
Denotational and operational semantics of programs
Descriptive complexity
Dynamic and algorithmic logics and schematology
Formal tools to describe concurrent computations
Lambda calculus and related topics
Foundations of logic programming
Logical foundations of data base theory
Logics for knowledge representation
Modal and temporal logics
Type theory in programming
Verification of programs

The Program Committee members were:

J.M. Barzdins (Latvian State Univ.)
J.Y. Halpern (IBM Almaden Research Center)
A.R. Meyer (MIT)
G.E. Mints (Inst. of Cybernetics Estonian Acad. Sci.)
A.L. Semionov (Scientific Council on Cybernetics USSR Acad. Sci.)
A.P. Stolboushkin (Program Systems Inst. USSR Acad. Sci.)
M.A. Taitslin, *Chair* (Kalinin State Univ.)
J. Tiuryn (Warsaw Univ.)

Arrangements were made for four invited lectures:

- Samson Abramsky, *Observational Logic and Process Semantics*
- Yuri L. Ershov, *Foundations of Semantical Programming*
- Yiannis N. Moschovakis, *A Mathematical Modeling of Pure, Recursive Algorithms*
- Gregory S. Tseitin, *Should Theory of Programming be Based on Predicate Logic?*

The program committee received forty-five abstracts and accepted twenty-two. Two were later withdrawn but one of the invited speakers was able to contribute a paper, and so this volume finally contains twenty-one full papers.

Final selection of papers was made at a meeting of the program committee in Tallinn, USSR, on December 14, 1988 attended personally by all the Soviet program committee members. The program committee members from outside the USSR were unable to attend, but they provided rankings and written comments which were fully taken into account in the final decisions.

The scope of the symposium was intended to be very broad, and in fact no submitted paper was rejected because it fell outside the scope of Logic in Computer Science. However, limits on the size of the program forced the committee to leave unselected many submitted papers worthy of publication. We would like to express our sincere thanks to everybody who submitted an abstract.

The papers in this volume represent many interesting trends in logical foundations of Computer Science, ranging from purely theoretical research to practical applications of theory. We hope the reader, like the symposium participants, finds the work reported here fruitful and stimulating.

A.R. Meyer, M.A. Taitslin
Cambridge, Mass.
February 1989

Table of Contents

Observational Logic and Process Semantics (abstract)

Samson Abramsky
Imperial College, London

We consider the question: why has the semantics of concurrency proved so much more elusive than that of sequential programming languages? One basic reason is that the traditional models of program input-output behaviour as functions or relations is no longer applicable in the presence of parallel composition; intermediate states also have to be considered. The question then arises: what should be regarded as observable behaviour of concurrent programs? No clear consensus has yet emerged on this point; instead, a considerable and still growing number of equivalences have been considered.

We attempt to develop some basic ideas which can be used to structure the discussion of such equivalences. We seek *both* a computational basis for process equivalences, *and* a uniform algebraic framework within which they can be described. For the first, we generalize the notion of *trace equivalence* (essentially the standard notion of behavioural equivalence in automata theory) by enriching the notion of atomic observation. We show that various natural choices of sets of atomic observations lead to a number of the best-known process equivalences. For the second point, we are able to cast our description of these observations, and the equivalences (and even, in some cases, the models) they induce, in the framework of *modules over quantales*, which provide a common generalization of topological spaces (and hence the usual structures of denotational semantics) and labelled transition systems, the basic structures used in the operational semantics of concurrency. Quantales can also be understood logically, as algebraic models of (non-commutative) linear logic in the sense of Girard. As such, they provide a new motivation for linear logic as the logic of testing, in which the state of the process being observed may change as a result of the observation.

A LOGIC FOR DATA DESCRIPTION

D.A.Arhangelsky, M.A.Taitslin
Kalinin State University
Zelabova 13
Kalinin 170013 USSR

INTRODUCTION.

The program languages use various data with different properties. For example, a data may be devided by the number system into decimal and binary, and by the representation mode into fixed-point and floating-point. In that consideration data are a set with equialent relations defined on this set. In [1] the language for data description is proposed and its semantics is described. Unfortunately the aim of this paper is not achived entierly in [1] since some formulas to be not valid are derivable in the proposed formal system. In this paper we build a new formal system and prove its correcteness and completeness with respect to the semantics proposed in [1], as well as resolvability.

1.THE LANGUAGE OF LOGIC DAL.

The expression of the language of logic DAL are built from simbols of the following pairwise disjoint sets:

VAR: propositional variables,

REL: relational variables, $\neg$, &, $\vee$, $\rightarrow$: classical propositional operations of negation, conjunction, disjunction and implication respectively, $\cup$, $\cap$: binary operations of union and intersection of relations respectively, [ ]: unary modal propositional operation, ( ): brackets.

The sets VAR and REL are nonempty denumerable sets.

The set EREL of terms is defined as follows:

1) if $x \in REL$, then $x \in EREL$;
2) if $x,y \in EREL$, then $(x \cup y),(x \cap y) \in EREL$;
3) $x \in EREL$ iff this follows from 1) and 2).

Now we define the set FOR of formulas of DAL as follows:
1) if $A \in VAR$, then $A \in FOR$;
2) if $A,B \in FOR$, then $\neg A$, $(A \vee B)$, $(A \& B)$, $(A \to B) \in FOR$;
3) if $A \in FOR$, $x \in EREL$, then $[x]A \in FOR$;
4) $A \in FOR$ iff this follows from 1),2),3).

To define the meaning of formulas of logic DAL we should fix a set OB of data items and a lattice of equivalence relations in OB, corresponding to properties of these data. By a model we mean a triple

M=(OB, R, m),

where OB is a nonempty set; S(OB) is the set of all the subsets of the set OB; R is a lattice of equivalence relations in OB, m: VAR $\cup$ EREL $\to$ S(OB) $\cup$ R is a function that assign subsets of the set OB and equivalence relations from R to the propositional variables and the relational terms respectively. m assign the union of items m(x) and m(y) in the lattice R to a relational term $x \cup y$, and assign the intersection of the items m(x) and m(y) in the lattice R to a relational term $x \cap y$.

Below the items of OB will be cold the points. We say that a formula A is satisfied by a point o in the model M (M,o sat A) iff the following conditions are satisfied:

M,o sat A iff $o \in m(A)$ for $A \in VAR$;
M,o sat $\neg A$ iff not M,o sat A;
M,o sat $A \vee B$ iff M,o sat A or M,o sat B;
M,o sat $A \& B$ iff M,o sat A and M,o sat B;
M,o sat $A \to B$ iff M,o sat $(\neg A \vee B)$;
M,o sat $[x]A$ iff for all $o' \in OB$ if $(o,o') \in m(x)$, then M,o' sat A.

2. Axiomatization.

Let t,x,y are terms, u is a relational variable, A,B are formulas; a expression of the form t(u/x) means the term obtained from the term t by replacing the relational variable u by the term x; a expression of the form A(u/x) means the formula obtained from the formula A by

replasing the relational variable u by the term x.

The theory T has the following schemes of axioms:

1) all formulas having the form of a tautology of the classical propositional logic,

2) $[x](A \to B) \to ([x]A \to [x]B)$,

3) $[x]A \to A$,

4) $\neg[x]A \to [x]\neg[x]A$,

5) $[t(u/x)]A \to [t(u/x \cup x)]A$,

6) $[t(u/x \cap x)]A \to [t(u/x)]A$,

7) $[t(u/x \cup y)]A \to [t(u/x)]A$,

8) $[t(u/y \cup x)]A \to [t(u/x)]A$,

9) $[t(u/x)]A \to [t(u/x \cap y)]A$,

10) $[t(u/x)]A \to [t(u/y \cap x)]A$,

and two inference rules:

$$\frac{A,\ A \to B}{B}$$ - modus ponens, and

$$\frac{A}{[x]A}$$ - necessition.

Lemma 1. If a formula A is provable in T then the formula A(u/x) is provable in T too.

The proof is obvious.

3. Completeness.

We denote the fact that $(p,q) \in m(x)$ by pxq.

We say that a identity $x \leq y$ holds for terms x and y, if pxq implies pyq for all the points p,q of all models.

Theorem 1. If a formula $[x]A \to [y]A$ is provable in T for some propositional variable A, then the terms x and y satisfied to the identity $y \leq x$.

Proof. Suppose there exists a model M and points p and q such that pyq, but not pxq. We build the new model M′ with the same basic set and equivalence relations. Put A is truth in a point o, if oxp, and A is false otherwise. Thus the formula [x]A is truth in p by definition. At the same time A is false in q, since p and q are not equivalent with respect to x. Hence, the formula [y]A is false in q.

On other hand, it is followed from provability of $[x]A \to [y]A$, that $[y]A$ is truth in p. In its turn, pyq implies that $[y]A$ is truth in q as well.

The theorem has been proved.

Lemma 2. Let $x \leqslant y$. Then there are terms x' and y' such that

1) $x' \leqslant y'$,
2) every variable appearing in x' appears in x' exactly once,
3) every variable appearing in y' appears in y' exactly once,
4) every variable appears in x' iff it appears in y',
5) the formula $[y]A \to [x]A$ is provable in T for all formula A if the formula $[y']A \to [x']A$ is provable in T for all formula A.

Proof. $x \leqslant y$ iff $(x \cup (x \cap y)) \leqslant (x \cap y)$. Let now u appears m times in x" and n times in y" for some $m,n \geqslant 1$. Introduce new variables u_{ij} (i=1...m, j=1...n). Replace the ith occurence of u in x" by $u_{i1} \cap \ldots \cap u_{in}$ for each i=1...m. Dually, replace the jth occurence of u in y" by $u_{m1} \cup \ldots \cup u_{mj}$. We obtain new terms x' and y' satisfied conditions 1)-4).

Suppose $[y']A \to [x']A$ is provable for all formula A. We fix a formula A. Let v_{ij} (i=1,...,m,j=1,...,n) be variables not appearing in A, x', y'. Replacing each occurence of u_{ij} in A by v_{ij}, we obtain A_1. The formula $[y']A_1 \to [x']A_1$ is provable in T. Replacing each occurence of u_{ij} in y' and x' by u, we obtain y" and x". By lemma 1, the formula $[y"]A_1 \to [x"]A_1$ is provable in T. By means axioms 7 – 10 the formulas $[x"]A_1 \to [x]A_1$ and $[y]A_1 \to [y"]A_1$ are provable. Thus the formula $[y]A_1 \to [x]A_1$ is provable, as well. Replacing each occurence of v_{ij} in A_1 by u_{ij} we obtain A. By lemma 1, the formula $[y]A \to [x]A$ is provable in T.

The lemma has been proved.

Theorem 2. If a identity $x \leqslant y$ is valid, then the formula $[y]A \to [x]A$ is provable in T for each formula A.

Proof. Let $x \leqslant y$ and every variable appearing in x apears in x exactly once. We prove that $[y]A \to [x]A$ is provable in T for each formula A.

We introduce a concept of series-parallel graphs (sp- graphs). By a graph G we mean a tuple $(V(G), E(G), i(G), e_{0G}, e_{1G})$, where V(G) is a set of vertices, E(G) is a set of edges, labeled by relational variables, i(G) is an incidence relation, e_{0G}, e_{1G} are distinguished vertices. If G is a graph then P(G) is the union of V(G) and E(G).

A sp-graph is defined recursively as follows.

An atomic sp-graph having two vertices e_0 and e_1, connected by a single edge labeled by some relational variable.

Let the intersection of P(G) and P(H) be empty.

A series connection of two sp-graphs G and H is obtained by means

identifying one of the distinguished vertices of the first graph with one of the distinguished vertices of the second graph. Other two distinguished vertices of the first and second graphs perform the distinguished vertices of the new graph.

A parallel connection of two sp-graphs G and H is obtained as follows. One of the disninguished vertices of the first graph is identified with one of the distinguished vertices of the second graph to form the first distinguished vertice of the new graph. In its turn, the second distinguished vertice of the first graph is identified with the second distinguished vertice of the second graph to form the second distinguished vertice of the new graph.

By a subgraph of a sp-graph G we mean a sp-graph $H=(V(H),E(H),i(H),e_{0H},e_{1H})$, where $V(H)\subseteq V(G)$, $E(H)\subseteq E(G)$, i(H) is the restriction of i(G) to E(H) x V(H), e_{0H},e_{1H} are in $\{e_{0G},e_{1G}\}$ or are only vertices incident with edges from E(H) and edges from $E(G)\setminus E(H)$.

We assign a set of sp-graphs G_u to each relational term u.

If u is a relational variable, then it corresponds to the atomic sp-graph with the edge labeled by the variable u.

If $u=u'\cup u''$, then u corresponds to the set of all the series connections of a graph from the set, corresponding to the term u', and a graph from the set, corresponding to the term u''.

If $u=u'\cap u''$, then u corresponds to the set of all the parallel connections of a graph from the set, corresponding to the term u', and a graph from the set, corresponding to the term u''.

We shall consider only terms with single occurrence of each variable.

We assign to each such term u the set of models M(G)=(OB,R,m), where OB is a set of all the vertices of a sp- graph G_u from the set, corresponding to the term u, R is the set of equivalence relations described below.

The equivalence relations are defined in M(G) as follows:

if u is a relational variable then puq means either p=q or p and q are connected by a edge labeled by the variable u;

if $u=u'\cup u''$ then puq means, that either pu'q or pu''q hold or there exists a point $o \in M(G)$ and subgraphs G', G'', G_1 of the graph G_u such that G_1 is a series connection of G' and G'', (p,o), (o,q), (p,q) are the couples of the distinguished vertices for G', G'', G_1 respectively, and either the relations pu'o and ouq in M(G'), M(G'') hold or the relations pu''o and ouq in M(G'), M(G'') hold;

if $u=u'\cap u''$ then puq means that both pu'q and pu''q hold.

It is easily to see that these relations are reflexive, symmetric,

and transitive. It is evident also that the relations corresponding to terms u'∩u" and u'∪u" are the greatest lower bound and the least upper bound respectively for the relations, corresponding to the terms u' and u".

We shall consider next operations on the terms, corresponding to the axioms 5-10.

1. Replacing of a subterm of form u∪u by the term u, that corresponds to replasing of two succesively connected subgraphs by one of its.

2. Replacing of a subterm u by the term u∩u, that corresponds to parallel duplicating of a subgraph.

3. Replacing of a subterm u by a term of the forms u∪v or v∪u, that corresponds to replacing of the subgraph G_u by the subgraph being the series connection of the graphs G_u and G_v.

4. Replacing of a subterm of the forms u∩v or v∩u by the term x, that corresponds to deleting from the subgraph some pathes between the distinguished vertices.

We are to prove if in a model M(G), corresponding to a term x, the relation pyq holds, where p and q are the distinguished vertices, then it is posible to obtain the term y from the term x by means the operations 1-4. Similarly it is posible to obtain every graph G_y from some graph G_x.

The proof is by induction on the summary length of x and y.

Case 1: y is a relational variable. The pyq holds in M(G). Hence, by definition, the points p and q are connected by the edge labeled the variable y in M(G). In accordance to the rule 4 we delete all the other edges and obtain the term y.

Case 2: y=y'∩y". Then the identies x≤y and x≤y" hold. First, in accordance with rule 2, we replace x by x∩x. Using the induction hypothesis, y' and y" are obtained from x by means rules 1-4. Emploing these conversions separately to both subterms, we obtin the term y'∩y"=y.

Case 3: x=x'∪x". Then the identities x'≤y and x"≤y hold. By induction hypothesis emploing the rules 1-4 to both x' and x" separately, we obtain the term y∪y and replace its by the term y in accordance with the rule 1.

Case 4: x is a relational variable, y=y'∪y". It follows either py'q holds or py"q holds. If we let the first, by induction hypothesis, y' is obtainable from x by means the rules 1-4.

Case 5: x=x'∩x", y=y'∪y".

We consider the case, when there exists such a point o that o