On Valentin Turchin's Works on Cybernetic Philosophy, Computer Science and Mathematics

Andrei V. Klimov[⋆]

Keldysh Institute of Applied Mathematics
Russian Academy of Sciences
4 Miusskaya sq., Moscow, 125047, Russia
`klimov@keldysh.ru`

Abstract. The history, context, main ideas, motivation and research objectives of Valentin Turchin's works on philosophy, cybernetics, computer science and mathematics are outlined. Valentin Turchin's scientific legacy comprises three parts: cybernetic philosophy with the notion of a metasystem transition as a quantum of evolution; application of the philosophy to analysis of evolution of human society; and application of the philosophy to science — cybernetics, computer science and mathematics. In computer science, his main contributions are the programming language Refal and program transformation technique known as supercompilation. In mathematics, he has developed a new constructive foundation of mathematics referred to as the Cybernetic Foundation of Mathematics.

Keywords: Cybernetic Philosophy, Metasystem Transition Theory, evolution, programming language Refal, supercompilation, Cybernetic Foundation of Mathematics.

1 Introduction

Valentin Fedorovich Turchin (14.2.1931–07.4.2010) was a scholar of a rare kind, for whom the purpose of scientific work was a derivative of his philosophical worldview and the meaning of life.

In this paper an attempt is made to give a bird's eye view of Valentin Turchin's scientific and philosophical legacy. His works on philosophy, society and Cybernetic Foundation of Mathematics are just briefed, while his achievements in computer science are discussed in more detail.

This is a slightly revised and updated version of the Russian paper [11].

[⋆] Supported by Russian Foundation for Basic Research grant No. 12-01-00972-a and RF President grant for leading scientific schools No. NSh-4307.2012.9.

2 Valentin Turchin's Biography in Brief

After graduation from the Physics Department of Moscow State University in 1952, Valentin Turchin worked at the Institute of Physics and Power Engineering in Obninsk, USSR. Shortly after defending his doctoral thesis on the physics of slow neutrons (which later became a book [17]), he was invited by academician M.V. Keldysh to the Institute of Applied Mathematics of the USSR Academy of Sciences and moved to Moscow in 1964. At that time, he also changed the direction of his scientific work from theoretical physics to philosophy, cybernetics, computer science, and programming. He expressed that theoretical physics was approaching its crisis, and to resolve it, we needed high automation of manipulation of physical and mathematical theories and formal linguistic models by means of computers. As a first step towards that great goal, in the 1960s he developed a programming language *Refal* for processing symbolic information, and in the 1970s laid the foundation for a new program transformation method, which he called *supercompilation*. In those same years a group of undergraduate and graduate students of Moscow State University, Moscow Engineering Physics Institute and the Moscow Institute of Physics and Technology started working with him on the implementation of Refal and supercompilation.

In 1972, Valentin Turchin joined the Central Research Institute of Automation Systems in Construction (TsNIPIASS), where he headed a laboratory. Together with his disciples, he continued to develop Refal and its applications, as well as supercompilation. In summer 1974, he was dismissed for his political and human rights activities (e.g., [15]) and advocacy of Andrei Sakharov and Alexander Solzhenitsyn. Before he and his family had to emigrate to the United States in 1977, he was unemployed and continued his scientific work at home. However weekly research seminars with his students on Tuesdays did not interrupt.

Upon arrival in the United States in 1978, Valentin Turchin worked for one and a half years in the Courant Institute of Mathematical Sciences in New York with Jacob Schwartz, the author of the SETL language. Thereafter, until his retirement in 1999, he was a professor at the City College of the City University of New York (CC CUNY). During these years he made his main contributions to supercompilation (see Section 6 below) and foundation of mathematics (Section 7), and continued development of cybernetic philosophy (Sections 8).

3 Valentin Turchin's Trilogy

In 1977, shortly before his departure from the Soviet Union, Valentin Turchin told about his conception of scientific and philosophical "trilogy", part of which he had already realized, and was going to work on the remaining areas:

The philosophical part comprises the cybernetic philosophy elaborated in his book "The Phenomenon of Science: a cybernetic approach to human evolution" [24]. He introduced the concept of a *metasystem transition* as a uniform quantum of evolution and illustrated it by many examples from the origin of the life to the emergence of the human being and then to human culture with science as its highest point. A metasystem transition is the emergence of a new level of control over already existing systems, which usually multiply, and construction of a system of a new kind comprised of the control subsystem and a variety of controlled subsystems. From Valentin Turchin's viewpoint, the evolutionary worldview also gives us an insight into moral problems what is good and bad as well as the meaning of life: "good" is what contributes to the evolution and creation, "bad" is destruction and inhibition of evolution. In the basis of the meaning of life lies the idea of creative immortality as a personal contribution to the eternal evolution.

The social part is application of the cybernetic philosophy to the evolution of society and social issues, presented in his book "The Inertia of Fear and the Scientific Worldview" [29]. Here he outlined his view of the history of the mankind, having separated it into three stages, based on the method of integration of people into society: physical coercion (slavery), economic coercion (capitalism) and liberal integration based on ideological "coercion", to which humanity is still going. To refer to that new social organization of the future, he used the old term "socialism". From his viewpoint, the transition from capitalism to socialism is a large-scale metasystem transition. As the evolution is essentially non-deterministic and metasystem transitions can occur in various ways, deadlocks and kickbacks are possible rather then stable forward movement to more and more complicate organization (as it is commonly assumed with the concept of "progress"). The socialism in the Soviet Union was an attempt to build a society, which integrates people based on ideas and spiritual culture, but it failed. Instead of proper socialism, the result was totalitarianism, in which individuals have lost their main trait — the free will, while a successful metasystem transition should create a new level of control to integrate subsystems in the system without destroying the basic features of the subsystems, but further developing them.

The scientific part is application of the concept of a metasystem transition to obtaining scientific results. Valentin Turchin told that a new philosophical ideas are impossible to promote without demonstrating their fruitfulness for specific scientific achievements. Only if a philosophy helps to produce new ideas and to analyze various phenomena in science and society, it will allure people. When Valentin Turchin talked about these plans in 1977, one of his achievements in the field of programming has been created, the second one was at the level of general principles and methods, and he created the third one later in the United States:

– the Refal programming language
– the method of program transformation referred to as supercompilation
– the Cybernetic Foundation of Mathematics

4 Refal as a Metalanguage

Valentin Turchin promoted the view of science as *formal linguistic modeling* of the reality, rather than learning absolute truths [6,24]. In particular, mathematics studies the properties of formal models rather than any ideal objects. With the advent of computers, the mankind has obtained a tool independent of the human mind, which worked with linguistic models. Since then, a person can avoid routine manual manipulation of symbols and place them on a computer, reserving to himself the development of algorithms and programs. The advent of computers and programming as a human activity is a large-scale metasystem transition in the history of mankind in general and science in particular, aimed at automating manipulation of formal linguistic models.

The next metasystem transition was automation of computer programming and emergence of programming languages and programs that produce programs, that is compilers. The programs themselves became objects of manipulation by programs.

In the mid 1960s, Valentin Turchin created the language Refal [18–20] as a meta-language for processing texts in formal languages to enable further development of this large-scale metasystem transition. In those days, Refal was a language of higher level than the existing languages for processing symbolic information like Lisp, Snobol, etc. From its inception to the end of the 1980s, Refal was actively used in the USSR for writing compilers, macro generators, interpreters, as well as for other kinds of text manipulation and transformation of linguistic models: computer algebra, artificial intelligence, construction and check of mathematical proofs, etc.

A major problem, which Valentin Turchin wanted to solve with the use of Refal and which is still open, is computer verification of mathematical texts from the Bourbaki treatise. Once in early 1970s, Valentin Turchin asked Sergei Romanenko, then a student of Moscow State University, to conduct experiments on expanding definitions in the formal language of the first volume of the Bourbaki treatise "Set Theory". However, even a very simple formula exploded exponentially in size. Being very surprised, he expressed an idea that some metasystem transition had to be performed, but then it was unclear how. Later he conceived supercompilation as a tool for performing such metasystem transitions (see Section 5 below). Although the problem of analysis of mathematical texts like that of the Bourbaki treatise is not solved yet, at least one promising approach that uses supercompilation has been identified [7].

In the area of programming languages, Valentin Turchin expected explosion of the number of domain-specific languages, according to the general principles of his *Metasystem Transition Theory*. In "The Phenomenon of Science" he formulated the *Law of Expansion of the Penultimate Level*, which says that when a new level of control emerges, subsystems of the prior level multiply quantitatively and/or qualitatively. In the 1960s and 1970s, we observed the development of formal and software means for description of languages, systems for programming interpreters and compilers (referred to as compiler compilers), macro generators, etc. This was the making of a metasystem over programming languages. The creation of new languages was facilitated, resulting in the growth of their number and complexity. Valentin Turchin emphasized that language creation is a natural human trait. Each application area must generate its own formal language. Considering the importance of this task, he demonstrated how to build a macro system using Refal as a macro language [23].

Interestingly, this prediction initially did not come true as he expected. In the 1980s, macro systems and compiler compilers went out of fashion. By the 1990s, it seemed that new programming languages ceased to appear. But then came the 2000s and construction of domain-specific languages became popular. As in the 1960s and 1970s, we have been observing multiplication of programming languages (e.g., scripting languages, domain-specific languages), data formats (e.g., XML-based ones), interface specification languages, and many other kinds of languages.

5 Emergence of Supercompilation

After the first applications of Refal for formal text processing, having practically observed their complexity, Valentin Turchin started finding ways to further automate the creation of efficient meta-programs, programs that generate and transform programs. Once at a seminar in 1971, he said: “We must learn how to manipulate computer programs like we manipulate numbers in Fortran.” Definitions of programming languages should become subject of transformations as well. This is a one more metasystem transition.

For Valentin Turchin, the Metasystem Transition Theory was not just conceptual framework, with which he analyzed phenomena, but also a “guide to action”. By learning how metasystem transitions happens, he revealed typical patterns and used them to intentionally construct new metasystem transitions in order to accelerate the development of scientific areas of interest to him. A lot of such material were supplied by metamathematics and mathematical logic. He paid much attention to these topics in his working seminars in Keldysh Institute in early 1970s. At that time he conceived supercompilation as a method of equivalence transformation of Refal programs [21, 22].

First Valentin Turchin expressed these ideas at a series of seminars in winter 1971–72. He wrote on a blackboard a simple program — an interpreter of arithmetic expressions in Refal, and suggested to perform “computation in general form” of a sample expression with variables instead of some numbers. He taught us that introduction of variables and computation in general form is a common metasystem transition used in mathematics, and we should take it for program manipulation as well. At that seminar, having manually performed computations over a function call with an arithmetic expression with variables, he got a text resembling the result of translation of the expression to computer code. This was what is now known as *specialization* of a programs with respect to a part of arguments.

Shortly thereafter, in 1972, he presented this technique in formal form [21]. This paper marks the beginning of the history of supercompilation. However, it contained its basic level only, called *driving* in the next more detailed publication in 1974 [22]. In the first paper [21], he also defined a universal resolving algorithm

(URA) based on driving. By a remarkable coincidence, the Prolog language, which is based on a similar technique, appeared in 1972 as well.

From 1974, being unemployed (for political reasons, as a dissident), Valentin Turchin could not publish papers, and the birth of supercompilation occurred in form of lectures at a series of 7 weekly seminars in the winter 1974–75, which took place in "ASURybProekt", Central Design and Technological Institute for Automated Control Systems of the USSR Ministry of Fisheries, where one of the permanent members of the seminar Ernest Vartapetyan worked.

All of the main ideas of supercompilation were presented in those lectures. In particular, at the third seminar, Valentin told how to produce a compiler from an interpreter as well as a compiler compiler by means of three metasystem transitions using a supercompiler. This was what later became known as *Futamura projections*. (Yoshihiko Futamura conceived these results before Valentin Turchin: the first two projections in 1971 [4], and the third projection — a compiler compiler — in 1973 [5].)

Supercompilation became a classical example of building a complex formal system by means of a number of metasystem transitions:

1. The first metasystem transition over computations: transition from concrete states to representation of sets of states in form of states with variables (referred to as *configurations*), from computation on ordinary values to computation with variables (referred to as *driving*), from computations along one thread to unrolling a potentially infinite tree representing all possible computation threads.
2. The second metasystem transition over driving: folding a (potentially) infinite tree into a finite graph by folding (looping back).
3. The next metasystem transitions over folding operations: various strategies to stop driving, performing generalization and reconstruction of the tree and graph. Valentin Turchin called the second metasystem transition and first versions of the third metasystem transition *configuration analysis*. (Notice that separation into metasystem transitions may be ambiguous, especially when they occurred simultaneously.)
4. Further metasystem transitions over computations and driving: *neighborhood analysis* — generalized computations with two kinds of variables (variable of the second kind are naturally called *meta*variables) in order to obtain not only the tree and graph of computations, but also a set of states and the set of configurations that go along the same way of concrete or generalized computations. Valentin conceived the neighborhood analysis to improve the configuration analysis as the next level of control, allowing to select better points of generalization and folding. (Later Sergei Abramov suggested other interesting application of neighborhood analysis, including testing [1].)

Prior to his departure from the USSR, Valentin Turchin was unable to publish anything on supercompilation proper. Only in a book on Refal [42], published in 1977 without attribution in order not to mention the name of Valentin Turchin, five pages (92-95) had been inserted in Chapter "Techniques of using Refal" at the end of Section "Translational problem", which contained formulas to generate a compiler and a compiler compilers.

A year before the departure, in Fall 1976, Valentin Turchin learned that Andrei Ershov conducted research into *mixed computation*, which has close objectives. When Andrei Ershov visited Moscow, they met in hotel "Academic" and exchanged ideas. They put into correspondence the Ershov's notion of a *generating extension* and Valentin Turchin's compiler generation formula. In [3], A.P. Ershov called the formula of the second metasystem transition that reduces a generating extension to a specializer (supercompiler), the *double run-though theorem by V.F. Turchin*. (Here "run-through" means "driving".)

6 Further Development of Supercompilation

In his first years in the United States in Courant Institute (1978-1979), Valentin Turchin described the ideas and methods on Refal and supercompilation in a big report [26]. Several of those ideas await further development and implementation still, e.g., walk grammars, the notions of *metaderivative* and *metaintegral* based on the walk grammars, metasystem formulas, self-application. The methods that deal with walks in the graph of configuration lie at the next meta-level compared to the *configuration analysis*, which deal with configurations. This work outlined the next metasystem transition to higher-level supercompilers. However, only recently the first two-level supercompiler [12] has been constructed, although based on other ideas. (Notice that this demonstrates the ambiguity and indeterminacy of metasystem transitions).

In the 1980s, working in CC CUNY together with a small group of students, he created and implemented on IBM PC a new version of Refal, Refal-5 [35], and developed a series of supercompilers for Refal. On those weak computers, they managed to carry out first experiments with supercompilers [25,32,43]. He also outlined approaches to problem-solving using supercompilers [27], in particular, to theorem proving [28].

The first journal paper on supercompilation [31], which is highly cited, was published in 1986.

Gradually researchers from other universities joined Valentin Turchin's work on supercompilation, most notably at the Department of Computer Science at the University of Copenhagen, chaired by Neil D. Jones, who has developed another method of program specialization — *partial evaluation*. However here we restrict ourselves to the line of Valentin Turchin's works and just mention some of the related papers of other authors.

In 1987, Dines Bjorner together with Neil Jones and Andrei Ershov organized a Workshop on Partial Evaluation and Mixed Computation in Denmark. At the workshop, Valentin Turchin presented a paper [34] on the first practical *whistle* — a criterion to terminate driving and generalize a configuration. The previous whistles were either too inefficient, or produce bad residual programs in automatic mode, or required help from the user.

In 1989, under "perestroika", Valentin Turchin resumed regular contacts with his Russian disciples. From then on, he visited Russia together with his wife almost every year, and intensive workshops were held in Moscow, Obninsk (1990) and Pereslavl-Zalessky.

In the late 1980s, he wrote a draft book on supercompilation. However, it remained unpublished, since he planed to improve it. The book contained the description of the second version of supercompiler SCP2. The copies of the book chapters were distributed among the participants of supercompilation workshop in Obninsk in summer 1990.

In the early 1990s, he returned back to the idea of building a more powerful supercompiler based on transformation of walk grammars. He restricted the class of grammars to regular expressions (compared to the Report-80 [27]), and presented the methods in paper [38]. However, they were not completed in form of algorithms or strategies and are still waiting for further development.

In 1994, Andrei Nemytykh worked almost a year with Valentin Turchin and they started development of the fourth version of a Refal supercompiler SCP4, which was completed by Andrei during next years [13,14].

In 1995, Morten Heine Sorensen and Robert Gluck suggested [16] to use a *well-quasi-order* on terms (*homeomorphic embedding*) as a whistle. Valentin Turchin liked this method and since then suggested to use it as a primary whistle in supercompilers. In particular, it was implemented in SCP4.

In 1996, Valentin Turchin wrote two last papers on supercompilation [40,41] with an overview of achievements and his vision of future work.

A comprehensive bibliography of his works on supercompilation may be found in [8].

Valentin Turchin always thought of putting supercompilation into practice. In 1998 he raised an investment and founded Supercompilers, LLC, with the goal of developing a Java supercompiler [10]. He was fond of this topic, but unfortunately the Java supercompiler has not been completed yet.

In recent years we observe a burst of interest to supercompilation, which Valentin Turchin dreamed about. The method of supercompilation has been polished, simplified and improved by many researchers. A number of experimental supercompilers for simple languages has been developed. But that's another story.

7 Cybernetic Foundation of Mathematics

Another great scientific achievement by Valentin Turchin lies in the foundation of mathematics. He was a constructivist in the spirit close to .. Markov. He said that no actual infinity exists. There are just language models and potentially infinite processes and enumerations. Like A.A. Markov, he sought to reduce all mathematical concepts to constructive algorithmic ones. He was not confused by the failure of the Markov's constructive mathematics to express all mathematical entities as algorithms. He saw the limitations of Markov constructive approach in that it is a closed system incapable of evolution. By expressing everything in terms of deterministic algorithms, we have a world in which metasystem transitions "degenerate", "saturated", cease to generate new quality.

Valentin Turchin managed to build an "open" constructive system by extending the world of algorithms with a model of the user of mathematics, a mathematician. This is a metasystem transition similar to the appearance of the concept of an observer in the modern physics. The wheels of his theory, referred to as Cybernetic Foundation of Mathematics, model multiple metasystem transitions. In this theory, he actually demonstrated what are formal metasystem transitions in action. As a result, he was able to give constructive interpretation of the notion of a set and the Zermelo-Fraenkel axioms based on the extended notion of an algorithm.

In 1983 he published a book on Cybernetic Foundations of Mathematics as a university report [30]. After that, he squeezed the material to two articles that have been accepted for publication in a journal, but with the condition that he would reduce them to a single article [33]. Unfortunately, the published version is badly readable due to excessive density of presentation. The full version is still awaiting its publication and translation to Russian.

It is naturally to assume that Valentin Turchin planned to combine the works on supercompilation and Cybernetic Foundation. He said he did not believe in the fruitfulness of the methods of machine theorem proving based on logic inference, and sought for constructive approaches. Algorithmic definitions of mathematical notions may become subject to manipulation and inference using methods similar to supercompilation. In turn, supercompilation can be enriched with effective means of constructing proofs of statements about programs.

8 Further Development of Cybernetic Philosophy

In the 1990s, Valentin Turchin continued the work on cybernetic philosophy together with Francis Heylighen and Cliff Joslyn. They founded the Principia Cybernetica Project [9] with a goal to elaborate various philosophic and scientific questions on the common cybernetic ground and to gather a community of researchers interested in these topics. Unfortunately, after about a decade of development the project became silent and wait for new enthusiasts.

In these years he also published several articles on the concept of the metasystem transition [39], cybernetic ontology [36] and epistemology [37]. Further discussion of these very interesting topics requires a separate paper.

9 Conclusion

The supercompilation history dates back for almost 40 years. Valentin Turchin planed to get a program transformation tool to implement his further ideas on manipulation of formal linguistic models. But the history moves much slower. Now that we have several experimental supercompilers, it is clear that it was impossible to implement such a system either on mainframes in the 1970s, or on personal computers in 1980s. But now with modern computers, supercompiler experiments became more and more successful. The time for such methods to go into wide programming practice has come. Then the great challenge of mass manipulation of formal linguistic models will come to the agenda, and someday these methods will lead to the burst of automated construction of physical theories, what Valentin Turchin dreamed about at the start of his scientific carrier.

Acknowledgements. I was very lucky to meet Valentin Turchin when I was a schoolboy and to belong to a group of students who started their scientific activity under his scientific mentorship. We became friends and professional team-mates for life.

I am very grateful to those of them with whom we still collaborate on the topics founded by our Teacher: Sergei Abramov, Arkady Klimov, Andrei Nemytykh, Sergei Romanenko, as well as to young researches who joined the community of Valentin Turchin's disciples and generate new ideas: Sergei Grechanik, Yuri Klimov, Ilya Klyuchnikov, Anton Orlov, Artem Shvorin, and to other participants of the Moscow Refal seminars.

It's a great pleasure for me to spend time together with Robert Glück, when we have a chance to meet and study Valentin Turchin's legacy and further develop his ideas.

We are very glad to meet and discuss various computer science topics with Neil D. Jones, who noticed Valentin Turchin's works at the early stage and whose deep scientific ideas and achievements have been influencing the works of our team very much.

Finally I express our love and gratitude to Tatiana Turchin for all she has been doing for us and for keeping alive the memory of Valentin Turchin.

References

1. S. M. Abramov. *Metavychislenija i ikh prilozhenija (Metacomputation and its applications)*. Nauka, Moscow, 1995. (In Russian).
2. O. Danvy, R. Glück, and P. Thiemann, editors. *Partial Evaluation. Dagstuhl Castle, Germany, February 1996*, volume 1110 of *Lecture Notes in Computer Science*. Springer-Verlag, 1996.
3. A. P. Ershov. On the essence of compilation. In E. Neuhold, editor, *Formal Description of Programming Concepts*, pages 391–420. North-Holland, 1978.
4. Y. Futamura. Partial evaluation of computing process – an approach to a compiler-compiler. *Systems, Computers, Controls*, 2(5):45–50, 1971.
5. Y. Futamura. EL1 partial evaluator (progress report). Internal report submitted to Dr. Ben Wegbreit, Center for Research in Computer Technology, Harward University, Cambridge, MA, USA, January 24, 1973.
6. R. Glück and A. V. Klimov. Metacomputation as a tool for formal linguistic modeling. In R.Trappl, editor, *Cybernetics and Systems '94*, volume 2, pages 1563–1570, Singapore, 1994. World Scientific.
7. R. Glück and A. V. Klimov. Metasystem transition schemes in computer science and mathematics. *World Futures: the Journal of General Evolution*, 45:213–243, 1995.

8. R. Glück and M. H. Sørensen. A roadmap to metacomputation by supercompilation. In Danvy et al. [2], pages 253–275.
9. F. Heylighen, C. Joslyn, and V. F. Turchin. *Principia Cybernetica Web*. `http://pespmc1.vub.ac.be`.
10. A. V. Klimov. An approach to supercompilation for object-oriented languages: the Java Supercompiler case study. In *First International Workshop on Metacomputation in Russia, Proceedings. Pereslavl-Zalessky, Russia, July 2–5, 2008*, pages 43–53. Pereslavl-Zalessky: Ailamazyan University of Pereslavl, 2008.
11. A. V. Klimov. O rabotakh Valentina Fedorovicha Turchina po kibernetike i informatike (on Valentin Fedorovich Turchin's works on Cybernetics and Informatics). In A. Tomilin, editor, *Proceedings of SORUCOM-2011*, pages 149–154, 2011. `http://sorucom.novgorod.ru/np-includes/upload/2011/09/05/15.pdf`. (In Russian).
12. I. G. Klyuchnikov and S. A. Romanenko. Towards higher-level supercompilation. In A. Klimov and S. Romanenkop, editors, *Second International Valentin Turchin Memorial Workshop on Metacomputation in Russia, July 1-5, 2010, Pereslavl-Zalessky, Russia*, pages 82–101. Ailamazyan Program Systems Instiute of RAS, 2010. `http://meta2010.pereslavl.ru/accepted-papers/paper-info-5.html`.
13. A. P. Nemytykh. *Superkompilyator SCP4: Obshchaya struktura (The Supercompiler SCP4: General Structure)*. URSS, Moscow, 2007. (In Russian).
14. A. P. Nemytykh, V. Pinchik, and V. Turchin. A self-applicable supercompiler. In Danvy et al. [2], pages 322–337.
15. A. D. Sakharov, R. A. Medvedev, and V. F. Turchin. A reformist program for democratization. In S. F. Cohen, editor, *An End to Silence. Uncensored Opinion in the Soviet Union from Roy Medvedev's Underground Magazine Political Diary*, pages 317–327. W. W. Norton & Company, New York, London, 1970. (Reprinted 1982).
16. M. H. Sørensen and R. Glück. An algorithm of generalization in positive supercompilation. In J. W. Lloyd, editor, *International Logic Programming Symposium, December 4-7, 1995, Portland, Oregon*, pages 465–479. MIT Press, 1995.
17. V. F. Turchin. *Slow Neutrons.* Israel Program for Scientific Translations, Jerusalem, Israel, 1965.
18. V. F. Turchin. Metajazyk dlja formal'nogo opisanija algoritmicheskikh jazykov (A metalanguage for formal description of algorithmic languages). In *Cifrovaja Vychislitel'naja Tekhnika i Programmirovanie*, pages 116–124. Sovetskoe Radio, Moscow, 1966. (In Russian).
19. V. F. Turchin. A meta-algorithmic language. *Cybernetics*, 4(4):40–47, 1968.
20. V. F. Turchin. Programmirovanie na jazyke Refal. (Programming in the language Refal). Preprints 41, 43, 44, 48, 49, Institute of Applied Mathematics, Academy of Sciences of the USSR, Moscow, 1971. (In Russian).
21. V. F. Turchin. Ehkvivalentnye preobrazovanija rekursivnykh funkcij na Refale (Equivalent transformations of recursive functions defined in Refal). In *Teorija Jazykov i Metody Programmirovanija (Proceedings of the Symposium on the Theory of Languages and Programming Methods)*, pages 31–42, 1972. (In Russian).
22. V. F. Turchin. Ehkvivalentnye preobrazovanija programm na Refale (Equivalent transformations of Refal programs). *Avtomatizirovannaja Sistema upravlenija stroitel'stvom. Trudy CNIPIASS*, 6:36–68, 1974. (In Russian).
23. V. F. Turchin. Refal-makrokod (Refal macrocode). In *Trudy Vsesojuznogo seminara po voprosam makrogeneratsii (Proceedings of the All-Union Seminar of Macrogeneration)*, pages 150–165, 1975. (In Russian).

24. V. F. Turchin. *The Phenomenon of Science: A Cybernetic Approach to Human Evolution*. Columbia University Press, New York, 1977.
25. V. F. Turchin. A supercompiler system based on the language REFAL. *SIGPLAN Not.*, 14(2):46–54, Feb. 1979.
26. V. F. Turchin. The language Refal, the theory of compilation and metasystem analysis. Courant Computer Science Report 20, Courant Institute of Mathematical Sciences, New York University, 1980.
27. V. F. Turchin. Semantic definitions in REFAL and automatic production of compilers. In N. D. Jones, editor, *Semantics-Directed Compiler Generation*, volume 94 of *Lecture Notes in Computer Science*, pages 441–474. Springer-Verlag, 1980.
28. V. F. Turchin. The use of metasystem transition in theorem proving and program optimization. In J. de Bakker and J. van Leeuwen, editors, *Automata, Languages and Programming*, volume 85 of *Lecture Notes in Computer Science*, pages 645–657. Springer-Verlag, 1980.
29. V. F. Turchin. *The Inertia of Fear and the Scientific Worldview*. Columbia University Press, 1981.
30. V. F. Turchin. The Cybernetic Foundation of Mathematics. Technical report, The City College of the City Univeristy of New York, 1983.
31. V. F. Turchin. The concept of a supercompiler. *Transactions on Programming Languages and Systems*, 8(3):292–325, 1986.
32. V. F. Turchin. Program transformation by supercompilation. In H. Ganzinger and N. D. Jones, editors, *Programs as Data Objects*, volume 217 of *Lecture Notes in Computer Science*, pages 257–281. Springer-Verlag, 1986.
33. V. F. Turchin. A constructive interpretation of the full set theory. *The Journal of Symbolic Logic*, 52(1):172–201, 1987.
34. V. F. Turchin. The algorithm of generalization in the supercompiler. In D. Bjørner, A. P. Ershov, and N. D. Jones, editors, *Partial Evaluation and Mixed Computation*, pages 531–549. North-Holland, 1988.
35. V. F. Turchin. *Refal-5, Programming Guide and Reference Manual*. New England Publishing Co., Holyoke, Massachusetts, 1989.
36. V. F. Turchin. The cybernetic ontology of action. *Kybernetes*, 22(2):10–30, 1993.
37. V. F. Turchin. On cybernetic epistemology. *Systems Research*, 10(1):3–28, 1993.
38. V. F. Turchin. Program transformation with metasystem transitions. *Journal of Functional Programming*, 3(3):283–313, 1993.
39. V. F. Turchin. A dialogue on metasystem transition. *World Futures*, 45:5–57, 1995.
40. V. F. Turchin. Metacomputation: Metasystem transitions plus supercompilation. In Danvy et al. [2], pages 481–509.
41. V. F. Turchin. Supercompilation: techniques and results. In D. Bjørner, M. Broy, and I. V. Pottosin, editors, *Perspectives of System Informatics, Second International Andrei Ershov Memorial Conference, Akademgorodok, Novosibirsk, Russia, June 25-28, 1996. Proceedings*, volume 1181 of *Lecture Notes in Computer Science*, pages 227–248. Springer, 1996.
42. V. F. Turchin, A. V. Klimov, A. V. Klimov, V. F. Khoroshevsky, A. G. Krasovsky, S. A. Romanenko, I. B. Shchenkov, and E. V. Travkina. *Bazisnyj Refal i ego realizacija na vychislitel'nykh mashinakh (Basic Refal and its implementation on computers)*. GOSSTROJ SSSR, CNIPIASS, Moscow, 1977. (In Russian).
43. V. F. Turchin, R. Nirenberg, and D. Turchin. Experiments with a supercompiler. In *Conference Record of the ACM Symposium on Lisp and Functional Programming*, pages 47–55. ACM Press, 1982.