A WEB-SYSTEM FOR COMPUTER EXPERIMENTS IN THE FIELD OF PROGRAM TRANSFORMATIONS

Margarita Knyazeva, Alexander Kleshchev

Abstract: The paper presents basic notions and scientific achievements in the field of program transformations, describes usage of these achievements both in the professional activity (when developing optimizing and unparallelizing compilers) and in the higher education. It also analyzes main problems in this area. The concept of control of program transformation information is introduced in the form of specialized knowledge bank on computer program transformations to support the scientific research, education and professional activity in the field. The tasks that are solved by the knowledge bank are formulated. The paper is intended for experts in the artificial intelligence, optimizing compilation, postgraduates and senior students of corresponding specialties; it may be also interesting for university lecturers and instructors.

Keywords: Knowledge bank; Ontology; Knowledge base; Ontology editor; Database editor; Knowledge processing; Program transformations; Optimizing compilation

ACM Classification Keywords: I.2.5 Artificial intelligence: programming languages and software

Introduction

The science, higher education and professional activity are closely linked in knowledge-intensive domains. Scientific achievements – the foundation of higher education content – are used in the professional activity and contribute to its progress. Higher education institutions train specialists both for science and professional activity. The more university graduates are aware of the latest scientific breakthroughs, the more they are in demand both in science and professional activity which is a validation and utility criterion of scientific knowledge in the last resort. It also formulates tasks to be solved by science. Finally, it sets up requirements for the training level of university graduates; the most important of which is a skill to apply obtained knowledge in practice. Progress in such domains is usually based upon ideas that help to solve problems in all the three areas – science, education and professional activity. One of the science-intensive domains where those areas are interlinked is computer program transformation.

At present the task of increasing processing power is still of current interest. The modern development of computer science is connected with new parallel architectures. With processors getting more powerful, the requirements to them are growing. The relevant software is necessary for achieving high efficiency of parallel machines. Computer architecture getting more complicated, the programming languages are also becoming more complicated. That leads to poor quality both of source programs and target ones. Therefore, to keep the gain obtained at the expense of the possibilities of new computer architectures it is necessary to optimize source programs and improve programming language compilers. A lot of problems connected with systems of program transformations still remain unsolved (for example, problems of taking context into account and choosing an order (strategy) of using transformations as any transformation used in wrong environment and at an inappropriate step may result in the deterioration of program instead of its improvement). Intensive scientific research on program transformations is to facilitate progress in high-technology professional activity – developing optimizing and unparallelizing compilers. Universities must train up-to-date specialists to ensure both scientific and professional activities in this field.

The main achievements in the three spheres linked with program transformations, major problems impeding progress in the filed and possible solutions are considered in the present paper.

Hereinafter classical transformations, restructuring transformations and transformations for parallel machines are regarded as program transformations. Classical transformations are those developed for sequential machines but also useful for parallel ones. Restructuring transformations are transformations that increase the degree of parallelism in the program at the expense of changes in the program structure. Transformations for parallel machines follow exclusively from the parallelism of the machine architecture [Evstigneev, 1996].

Some basic notions and scientific achievements in the field of program transformations are considered in the following part.

This paper was made according to the program of fundamental scientific research of the Department of Power Engineering, Mechanical Engineering, Mechanics, Control Processes, the Russian Academy of Sciences, within the framework of the project “Synthesis of intellectual systems of control over knowledge bases and databases”; according to the program of the Far Eastern Branch of the Russian Academy of Sciences (the project code is 05-III-ZH-01-008).

Modern State on Program Transformations

The necessity of program optimization was first realized almost together with the creation of the first translators of programming languages. The practice showed that raised level of a source programming language told negatively on object program efficiency obtained as a result of translating. In this connection at the end of 1950s the task to increase efficiency with the help of transformations during the process of translating was taken on. This task will be called *program optimizing transformation task.* Transformations that increase the program efficiency are called *optimizing transformations* (OTs). *The system of optimizing transformations* is a set of OTs together with the strategy of their utilization [Kasyanov, 1988].

Program models provide theoretical base for studying and substantiating algorithms of program transformations. Thus, at present there are models of both sequential and parallel programs in the field of program transformations. Program transformations are introduced in terms of program models, i.e. over schemas not programs. The transformation description consists of two parts – the description of *contextual conditions* of the transformation, i.e. conditions under which the application of the transformation is possible and preferable, and the description of the *transformation* itself that assigns what should be changed in the schema. The same transformation under various contexts can determine transformations different in their contents. Each transformation is described in terms of one model and its application, as a rule, does not exceed its limits. It makes it possible to consider the transformed schema as a source one for further transformations. The transformation context is described in terms of program model and flow analysis, which is a means of getting reliable and accurate information about program performance without its execution. There are various classifications of transformation that are reflected in catalogues and information systems of transformations. There are means of formalizing knowledge about program transformations. One can state that program transformations are a developed field of the applied science that has already accumulated/acquired/gained extensive knowledge.

The knowledge about program transformations is necessary for using it in the professional activity while developing optimizing compilers. At present, there are compilers with wide sets of transformations for most widespread programming languages and types of computers. Thus, optimizing compiler manufacturing is a practical application of program transformations.

Scientific and practical activities connected with program transformations require training relevant specialists. Some of these specialists perform practical tasks of developing optimizing compilers; others conduct research on program transformations. Thus, specialists in the sphere of optimizing compilers are still needed both in science and practice. They are trained by leading universities of Russia, the Commonwealth of Independent States (Former Soviet Union), and other countries. Knowledge of modern scientific achievements and ability to develop optimizing compilers are the essential requirements for the level of their training that is supported theoretically, methodologically, and instrumentally.

However, there are a few interwoven problems in science, practice and education in the field of program optimization that are analyzed in the following part.

Analysis of Problems and Possible Ways of Their Solving

The main problem in the sphere of program optimization science is that it is impossible to carry out computer experiments opportunely. Their goal is to define how often transformations can be applied in real programs and what effect can be achieved. The only means of conducting such experiments is optimizing compilers. However, the period between the moment of the publication of the new transformation description and the moment of the ending of realization of optimizing compiler containing this transformation is so long that the results of computer experiments with this transformation appear to be out-of-date. Besides, an optimizing compiler usually contains a wide set of transformations and built-in strategy of their application so it is quite problematic to get results of computer experiments related to one transformation (not to the whole set).

The heterogeneity of notions and models in the sphere of program transformations, impossibility to prototype optimizing compilers and their quick moral aging are important problems.

The object of transformation is modeled by a lot of various schemas that are described in different terms, which makes it difficult to select one program model when designing each optimizing compiler. Describing transformations authors use different systems of notions, which hampers reading scientific literature by optimizing compiler developers.

Optimizing compiler is an extremely complicated program system the development of which is quite labor-intensive and time-consuming. Real characteristics of such a compiler are not quite know until it is used. Usually when developing it they prototype it to determine its characteristics. However, no prototyping facilities for optimizing compilers have been proposed yet which makes the risk higher.

Finally, each optimizing compiler is a program that is difficult to modify, i.e. it is virtually impossible to change a set of transformations in it. Because of a long period of its development, the compiler becomes somewhat obsolete by the moment of its putting into operation in comparison to the current level of program optimization science and gets more obsolete during its utilization.

The above-mentioned heterogeneity of notions and models in the sphere of program transformations and impossibility to use active teaching methods in education are major problems.

This heterogeneity creates difficulties for teachers when they systematically give information on the results achieved in this sphere and for students when they apprehend it.

Training students needs acquiring logically complicated theory and gaining practical skills of using its results. Special tools are necessary to let students do practical tasks and carry out research sufficiently within the time limits set by the curriculum. However, such tools have not been designed yet. Therefore, as a rule, students try to assimilate program transformations as a theoretical discipline gaining practical skills after their graduation from the university.

Thus, in spite of significant scientific, practical and educational achievements in program transformations there are a few problems hampering their development. To solve the above-mentioned problems is a topical task. The multipurpose computer knowledge bank is used as the general concept within the framework of which these problems can be solved [Orlov, 2003a].

The following section considers Specialized Knowledge Bank on Program Transformations (SKB_PT) as the concept of program transformation information control to solve the problems [Orlov, 2003a].

Concept of Specialized Knowledge Bank on Program Transformations

The general tasks of SKB_PT are centralization of knowledge on program transformations, coordination of their processing and collective development in order to achieve the most quality and up-to-date knowledge in this sphere and facilitate its using in science, education and professional activity.

The definition of Multipurpose knowledge bank (MKB) is given in the work [Orlov, 2003a]. According to it, MKB is a set of specialized knowledge banks (SKB). SKB for support of scientific research, educational and professional activities in a domain is a resource integrating the relevant information, providing specialists and computer programs with the access to this information, and containing tools for performing those tasks of information processing the effective methods of solving them have been already developed.

SKB_PT consists of Information Content (IC), Shell of IC, Program Content (PC) and Administration Block (AB). IC contains the relevant information. Shell of IC provides computer programs with the access to the information. Editing tools are necessary to form and develop IC. The work [Orlov, 2003b] proposes the concept of universal editor of information of different integration levels (Editor of IDIL). Other tools of information processing can be added to PC as effective methods of solving the corresponding tasks are developed. AB of SKB_PT manages users and controls the life cycle of SKB_PT. A special user of SKB_PT – Administrator – performs functions of managing other users and information resources. Shell of IC can be built on; new programs can be developed for PC. A special user of SKB_PT – Supporter – is responsible for building on Shell of IC and adding new programs to PC of SKB_PT [Orlov, 2003a]. The work [Orlov, 2003c] describes general methods of realization of MKB and specialized knowledge banks.

The information contained in IC of SKB_PT includes the ontology of knowledge on program transformations, ontologies of programming languages, the ontology of Structural Program Models (SPM), the program base storing source programs represented in the ontology of SPM, the knowledge base storing knowledge about program transformations, the archive of program transformation histories.

The editing tools are the editor of the ontology of knowledge on program transformations, ontologies of programming languages and the ontology SPM, the specialized editor of database and editors of programming languages that are developed using the editor of IDIL.

The users of the editing tools are carriers of the following information: ontologies of programming languages (linguists), the ontology of SPM, the ontology of knowledge on PT (knowledge engineers), knowledge on a domain (experts) and programs (programmers).

Besides information carriers, other users solving other IC tasks (connected with program transformations) can work with SKB_PT: scientists, optimizing compiler developers, teachers, students (users solving training IC tasks), and guests (users that are allowed only to view IC).

PC of SKB_PT (i.e. service programs realized through the shell) includes editing tools of SKB_PT, Program transformer of SKB_PT, Tools visualizing program transformation histories and Code generators on different platforms. The prototyping tool for optimizing compilers (PT_OC) is also a part of PC. When entering, Program transformer gets the structural program model (the object of transformations), the data about necessary transformations from the database; when the processing is over it gives the transformed model of structural program and the information about the applied transformations – the program transformation history. The visualizing tools make it possible to view the history. Code generators on different platforms let generate object programs according to the transformed models of structural programs. The prototyping tool for optimizing compilers integrates four subsystems into one system (the optimizing compiler prototype): Program transformer, editors of programming languages, visualizing tools for the transformation history and Code generators on different platforms.

The following section considers ways of using SKB_PT for scientific, industrial and educational purposes in the field of program transformations.

Possible Usage of Specialized Knowledge Bank on Program Transformations for Scientific, Industrial and Educational Purposes

The proposed concept of specialized knowledge bank allows to support the collective development of information resources (first of all, databases) and process them with the help of computer programs.

The specialized knowledge bank can be used to support scientific research. It allows to minimize labor costs when writing scientific reviews, to make it possible to classify optimizing transformations, including new ones, to form and develop notion systems in this area, to include new transformations in the knowledge system, to promptly introduce specialists to new results, to conduct computer experiments, to present scientific results in a form convenient to use in the professional activity, to compare new results with the ones archived before.

Having decided to participate in the activity of SKB_PT, the scientist applies to Administrator of the bank for a registration. In the application he\she explains which class of users he/she wants to belong to, what tasks he/she would like to solve, informs of his/her qualification in the sphere of program transformations. After screening the application (and, possibly, consulting the scientific society in order to get approval of his/her qualification and whether granting his/her application is desirable), Administrator makes a decision to register the applicant as a bank user and informs him/her of it via e-mail. Administrator gives Editing tools to scientists and creates theories for editing in the experimental domain of Information content. Administrator ensures that the information being edited by one user will be unavailable to other users both for editing and using. After describing and modifying the given theories, scientists submit an application to store them in the effective domain of IC. Administrator analyses the theories and makes a decision whether to provide free access to them or send them to be improved.

Together with the application to open the edited theory in IC for free access, the scientist can give his/her articles (monographs, textbooks, etc.) describing it. That helps Administrator and other scientists to analyze the theory. When opening the modified theory for free access, Administrator makes a decision if it should substitute the old one (in case of its existence) or be left as an alternative. The base theories in the knowledge bank are the ontology and program transformation database [Artemjeva, 2002a] [Artemjeva, 2003b] [Artemjeva, 2003c]. Specialists can develop their own theories using the base ones as a foundation or propose their variants for storage and collective exchange.

Program transformation experiments can be conducted in the following way. When entering, Program transformer of the knowledge bank gets SPM. SPM can be designed with the help of high-level language editors and the program transforming the results of the work of the editors into the structure determined by the SPM ontology. SKB_PT Program transformer makes a control flow analysis and data flow analysis in SPM. The obtained information is a source for transformation modeling. First, Program transformer defines saving sectors of SPM on the base of contextual conditions; then it transforms SPM. Transformations are made on the base of the transformation knowledge given in the knowledge database. The transformed SPM is a result of work of SKB_PT Program transformer. Special code generators transform SPM into a representation necessary for further processing (for example, imperative one). Measuring systems on different platforms allow knowing experimental results of the transformation efficiency. Visualizing tools provide information on histories of applied program transformations.

SKB_PT can be used for prototyping of optimizing compilers. The specialist applies to Administrator of the bank for a registration. Administrator provides him/her with prototyping Tool, with the help of which the specialist integrates three subsystems of the bank: Programming language editors, Program transformer of the knowledge bank and Code generators on different platforms. The prototype developer selects a language of a number of languages having editors in the bank, a code generator on the necessary platform of a number of code generators in the bank, a set of optimizing transformations from the database and assigns the strategy of applying these transformations. If an empty set of optimizing transformations is assigned, the compiler prototype will be non-optimizing. Optimizing compiler prototyping makes it possible to research strategies and transformation sets in such compilers.

The specialized knowledge bank can be used for self-developing optimizing compilers: a set of optimizing transformations contains all the transformations from the database both in its current state and in all the future ones. Since the database is constantly modified – new transformations are added and morally aged ones are excluded, characteristics of the compiler prototype are automatically changed according to the changes in the database.

SKB_PT can also be used for training students. Teachers can use Information content of the bank in their preparation for lectures on program transformations. Laboratory tutorials on program transformations can be given on the base of the specialized knowledge bank. Students are to prototype optimizing compilers and conduct experiments with them, to replenish the bank database. They are supposed to find new optimizing transformations in the scientific literature, to include them in the database, carry out research. Students are to use scientific publications on program transformations to find necessary knowledge to fulfill the task, to formalize the knowledge, and to put it into the database by means of the knowledge editor. To conduct the experiment the student has to select a lot of structural programs in the source programming language as an experimental material, to put these programs into the bank Archive by means of language editors and get optimized versions of the programs by means of Program transformer. Students can get transformation histories of these programs and study them using the bank visualizing tools.

Students will better understand the subject – program transformations – by solving tasks and carrying out research. The bank mission is to provide thorough feedback in training and give an opportunity to acquire practical skills in knowledge formalizing and using; the teacher's role is to estimate the final result.

Conclusion and Acknowledgements

This paper reviews the present of scientific research, professional activity and education and analyses the problems in the area of program transformations. The information resource concept based on the modern paradigm of information computer processing is proposed as an approach to the problems. The introduced resource is called Specialized Knowledge Bank on Program Transformations. The classes of users, the structure of its information and program content are described. The paper also describes the possible usage of knowledge banks in scientific research in the industry and education.

Bibliography

[Artemjeva, 2002a] Artemjeva I.L., Knyazeva M.A., Kupnevich O.A. Domain ontology model for the domain "Sequential program optimization". Part 1. The terms for optimization object description. In The Scientific and Technical Information, 2002. (In Russian).

[Artemjeva, 2003b] Artemjeva I.L., Knyazeva M.A., Kupnevich O.A. Domain ontology model for the domain "Sequential program optimization". Part 2. The terms for optimization process description. In The Scientific and Technical Information, 2003. (In Russian).

[Artemjeva, 2003c] Artemjeva I.L., Knyazeva M.A., Kupnevich O.A. Domain ontology model for the domain "Sequential program optimization". Part 3. Examples of optimizations transformation description. In The Scientific and Technical Information, 2003. (In Russian).

[Evstigneev, 1996] Evstigneev V.A., Kasyanov V. N. Optimizing transformations in unparallelizing compilers. Programming, 1996, № 6, pp. 12-26. (In Russian).

[Kasyanov, 1988] Kasyanov V. N. Optimizing transformations of the programs. Moskow: Nauka, 1988. (In Russian).

[Orlov, 2003a] Orlov V.A., Kleshchev A.S. Multi-purpose bank of knowledge. Technical report. Part 1. Notion and policy. Vladivostok: IACP FEBRAS, 2003. 40 p. (http://www.iacp.dvo.ru/es/) (In Russian).

[Orlov, 2003b] Orlov V.A., Kleshchev A.S. Multi-purpose bank of knowledge. Part 3. A notion of a unified idea editor. Vladivostok: IACP FEBRAS, 2003. 28 p. (http://www.iacp.dvo.ru/es/) (In Russian).

[Orlov, 2003c] Orlov V.A. Multi-purpose bank of knowledge. Part 6. Implementation details. Vladivostok: IACP FEBRAS, 2003. 28 p. (http://www.iacp.dvo.ru/es/) (In Russian).

Authors' Information:

Margarita A. Knyazeva, Alexander S. Kleshchev, – Institute for Automation & Control Processes, Far Eastern Branch of the Russian Academy of Sciences; 5 Radio Street, Vladivostok, Russia

e-mail: mak@imcs.dvgu.ru, kleschev@iacp.dvo.ru