Key Personal Participated in the Current Project Financed by Rustaveli Foundation (2015-2017 years):The Head of the Project – Jemal Antidze

CLP(HC) : Foundation, Implementation, Applications

1. Novelty of research, goals and objectives

1.1 **Topicality of the problem and research novelty**

Declarative Programming is a style of programming that concentrates on what to do, rather than how to
do it. Some examples of declarative programming paradigms are:
(1) Functional programming;
(2) Logic programming;
(3) Rule based programming.

Constraint logic programming [JM94, JMM+98] is one of the most successful areas of logic programming, combining logical deduction with constraint solving. Solid theoretical foundations, efficient implementations, and a wide range of academic and commercial applications make constraint logic programming systems popular tools.

In logic programming [Lyo87], computations are done with the help of a special logical inference method, SLD-resolution. At each step of the inference, instantiations of variables are computed by the unification algorithm, which is a (constraint) solving method for equations in a free algebra of terms.

Constraint logic programming generalizes this approach in two ways: First, the constraint domain is a parameter that can be replaced by concrete domains (for instance, by real closed fields, finite domains, Presburger Arithmetic, algebras of finite or rational trees, etc.); Second, constraints may consist of not
only conjunctions of equations (as it is in the unification case), but also of more complex formulas containing equations, inequalities, disequations. At each step, satisfiability of the constraints is checked. Full constraint solving and answer computation is done only in the last step (if needed).

An important research direction in constraint logic programming is introduction a new constraint domain, designing an efficient satisfiability and

solving procedure for it, and putting it in the general constraint logic programming framework. As examples, we could mention CLP(R) [JMP+92], CLP(FD) [CD96], and RISC-CLP(Real) [Hon93]. Some of the members of our project team contributed in the development of constraint logic programming over hedges CLP(H) [DFK+14].

However, there is no CLP which allows hedges and contexts at the same time in the language.

A calculus for rule-based programming over hedges and contexts has been introduced in [MK06]. Its prototype implementations and applications in XML-related topics have been described in [DMK09, CDF+10, DK10]. However, constraint logic programming over hedges and contexts is more general than the above mentioned variant of rule-based programming, and it practically has not been studied.

The domain we are going to study in this project is a combination of hedges and contexts in a single framework. Hedges gained popularity in recent years because of interesting applications: They naturally model XML data, program schemata, ambients, multithreaded recursive program configurations where the number of parallel processes is unbounded, etc. Hedges has an applications in rewriting [Ham97], knowledge representation [Men11], theorem proving [kut02] and contexts has an applications in semantics of natural language [Ant06, Ant07, Kol98], program analysis and transformation[GT07], just to name a few. It should be also noted that the programming language of Mathematica [Wol03] support programming with hedge variables (placeholder for hedges) and an extension of Haskell programming language [Moh96] supports programming with context variables (placeholder for contexts). Therefore, including them into the framework of constraint logic programming could lead to an interesting and useful extension.

Solving equations between hedges and contexts is challenging task in unification theory, since both hedge unification and context unification are in the PSPACE class. Therefore, research on solving equations over compressed terms and incorporation into the framework proposed is an interesting and useful task. Hence, we have to find a good trade-off between expressiveness of constraints and their efficient solvability.

The main novelties will be:

1. Studying constraint logic programming over hedges and contexts.

2. Designing a constraint solver for theories over hedges and contexts.
3. Incorporating solving techniques for compressed terms into the domain hedges and contexts.
4. Implementation of programming system and its applications.

1.2 Research subject and objectives

The research subject is Foundation, Implementation and Applications of constraint logic programming over the domain hedges and contexts. More specifically, our goal is to introduce constraint satisfiability checking and constraint solving methods for hedges and contexts and to incorporate it into the general framework of constraint logic programming.

Constraints that we consider involve terms that are built over unranked function symbols and may contain four kinds of variables: for terms, for hedges, for function symbols, and for contexts. Term and hedge variables are first-order ones. Function and context variables are of second-order. A context variable applies to a single term. It can apply neither to a hedge variable (that is not considered to be a term) nor to a hedge. A substitution is a finite mapping defined on variables. It maps a term variable to a term, a hedge variable to hedges (to a finite sequence of terms and hedge variables), a function variable to a function symbol or a function variable, and a context variable to a context. The latter is a term with a single occurrence of a special constant 'o', called the hole. A context C can apply to a term t, resulting into a term $C[t]$, which is obtained from C by replacing the hole in it with t. In terms, different occurrences of the same unranked function symbol may have different number of arguments. These notions and operations are best illustrated by examples:

- A term: *f(a, f(Xt, Xc(b)), Xf(a, Xf(b), Xs)).* Here *f,a,b,c* are function symbols, *Xt* is a term variable, *Xc* is a context variable, *Xf* is a function variable, and *Xs* is a hedge variable.
- A context: *g(g(a),* o*, c).*
- A substitution: {*Xt* →*f(a, Yt), Xc* →*g(g(a),* o*, c), Xf* →*h, Xs* →*(f(a), Xt, h, Ys)*}.
- Application of a substitution to a term:
  - Term: *t=f(a, f(Xt, Xc(b)), Xf(a, Xf(b), Xs))*
  - Substitution: σ = {*Xt* →*f(a, Yt), Xc* →*g(g(a),* o*, c), Xf* →*h, Xs* →*(f(a), Xt, h, Ys)*}
  - Applying the given substitution to the given term gives a term:

tσ *=f(a, f(f(a, Yt), g(g(a), b, c)), h(a, h(b), f(a), Xt, h, Ys)).*

An unranked alphabet and four different kinds of variables give the language flexibility and strong expressive power, but it makes constraint solving a difficult task.
Besides equations and disequations, our constraints may contain membership atoms and their negations. These atoms restrict possible values of hedge variables by a regular hedge language, and possible values of context variables by a regular context (tree) language.
To illustrate the expressive power of CLP(HC), we describe how the general rewriting mechanism can be implemented in the framework proposed in this project:
rewrite(*Xc(Xt), Xc(Xty)*) ← **rule**(*Xt, Xty*)
where *Xt* and *Xty* are term variables, and *Xc* is a context variable.
It is assumed that there are clauses which define the **rule** predicate. The rewrite clause says a term *Xc*(*Xt*) can be rewritten to *Xc*(*Xty*) if there is a rule such that **rule**(*Xt*, *Xty*) succeeds.
An example of the definition of the **rule** predicate is:
rule(*Xf*(*Xs_1*,*Xs_2*),*Xf*(*Xs*y)) ← *Xs*_1 ∈ f(a*) · b*, *Xs*_1 = (*Xt*,*Xs*z), *Xs*y = (*Xt*,f(*Xs*z)), where the constraint *Xs*_1 ∈ f(a*) · b* requires *Xs*_1 to be instantiated by sequences from the language generated by the regular sequence expression f(a*) · b* (that is, from the language {f, f(a), f(a,a), ... , (f,b), (f(a),b), … , (f(a, ... ,a),b, ... ,b), ...}.
With this program, the query ← rewrite(f(f(f(a,a),b)), *Xt*) has two answers: { *Xt* →f(f(f(a,a),f))} and
{*Xt*→f(f(f(a,a),f(b)))}$.
This examples illustrates the benefits of all four kinds of variables. In particular, this variables make the language powerful and flexible. They help to traverse a (tree representation of a) data term uniformly both in the horizontal and vertical directions, in one or in arbitrary many steps. Moreover, it is possible to constrain hedge and context variables with regular languages. One can also notice, that the obtained code is pretty compact and declaratively clear.
In many applications, data terms can be huge. An instance of such an application can be XML processing, where an XML documents might contain tens or hundreds of thousands of nodes. To deal with space problems, various term compression techniques have been introduced, see, e.g., [BGK03,

BLM08]. Respectively, there is a need in solving techniques over compressed terms directly, without decompressing them [GGS+08, GGS+09]. We will investigate how these algorithms can be incorporated in CLP(HC). It can also be that we need to develop a special algorithm for our purposes
for the encoding in [BGK03], that uses integer exponent to encode unranked trees, without binarizing them.
CLP can be seen as a generalization of logic programming, its semantics is compatible with semantics of logic programs and hence prolog is a natural choice to implement the CLP(HC). The obtained system will have rich capabilities for various applications. First, the system will be suitable for XML-related tasks, because of its ability to work with unranked terms and with compressed terms. It will also have useful applicability in membrane computing [Pau] and computational linguistics.
The project objectives correspond to solving problems described in this section and can be formulated as the following four tasks:
Title of the task Expected time needed for accomplishment of the task, in months
Key personnel
1. Designing decision and solving procedures for hedges and contexts with membership constraints. Proving termination, soundness, and completeness theorems.
1-12 **months: Jemal Antidze, Besik Dundua, Irakli Kardava**
2. Integration the solving procedure into CLP and study operational, declarative and fixpoint semantics of obtained framework
7-12 **months: Jemal Antidze, Besik Dundua, Irakli Kardava**
3. Incorporating solving techniques for compressed terms into CLP(HC)
13-18 **months: Jemal Antidze, Besik Dundua, Irakli Kardava**
4. Implementation of the CLP(HC)
19-21 **months: Jemal Antidze, Besik Dundua, Irakli Kardava**
5. **Applications**, preparation of the documentation
22-24 months: Jemal Antidze, Besik Dundua, Irakli Kardava
 Count months from the starting date of the project.
 Research methods and expected outcomes
2.1 ***Compliance of research methods with the objectives of the project***

Among the members of the project team there are international experts with many years of experience in studying semantics of programming languages and designing and implementing algorithms for solving and deciding various kinds of constraints. The methods, that the members of the team will use, are the standard ones for defining semantics of constraint logic programming, designing constraint solving procedures and Implementation. Some members of the team studied semantics of constraint logic programming over hedges [DFK+14]. Our methods will be based on the ones used there. They will be extended to accommodate techniques that deal with context variables. The constraint solving procedures will be formulated in a rule-based manner, which is considered to be the de-facto standard way of designing such procedures. The rules should transform constraints into constraints. For procedures formulated in a rule-based way it is easier to define a complexity measure and to prove properties (termination, soundness, completeness). Moreover, the techniques of implementation CLP is also quite developed. Research on solving equations over compressed terms is new. We will have to study how we can incorporate the existing methods [GGS+08, GGS+09] in our framework.

2.2 *Expected outcomes of the research and their significance for scientific direction/directions of the research*

Expected results are both of theoretical and practical character. We plan to construct new solving procedures for special classes of constraints over quite a rich a flexible term language. This language contains unranked function symbols, four different kinds of variables. The classes of constraints will be defined in such a way that we can prove termination, soundness, and completeness of the procedures for them. Moreover, we will show for which kind of clauses and goals the generated constraints fall into these classes.

We see potential applications of the system developed in the project mainly in three areas: Web-related applications, software engineering, and education. One can possibly experiment with its use also in bioinformatics (membrane computing, P systems) and computational linguistics (linguistic querying). Uniqueness and competitiveness of the tool is based on the combination of the following aspects:

Flexible and intuitive syntax based on unranked terms, four kinds of variables, and well-known logic programming notation, that enables users to write compact and declaratively clear code.
Powerful constraint solving module and the mechanism to work with compressed terms that enables the system to tackle problems of significant size.
Usage of unranked symbols that is particularly appealing for XML-based applications as well as for bioinformatics (membrane computing).
In addition, the group members plan to demonstrate the tool in the programming and computational logic classes they currently teach.
The table below shows interim expected outcomes/countable indicators for each reporting period:

For each reporting period: Countable indicators for expected outcomes of the accomplished work

Add a table if necessary.

(***Countable Indicator****: One or several countable interim outcome/s achieved after accomplishment of each task in reporting period/stage. For example, 2 scientific articles will be published; 3 conference*
theses will be prepared; specific (name) block-scheme of experimental device will be constructed, etc.).

 Count months/periods from the starting date of the project.

 Management of the Project

As from the project's tasks is clear, we must fulfill a large volume of the works. This demands hard works of the participants to complete with success the project. Two participant are young specialists, but they have sufficient experience to complete the project with success. They are experienced programmers, which have resolved some important problems by computer. They have the experience to work in projects. The manager is experienced specialist in the domain (See his CV). We are sure, that this project will be terminated successfully. J.Antidze will manage all tasks. All participants will take part in achievement of all tasks. We must buy two powerful personal computers to use them as at the working place as well at home to estimate correctly the speed of experiments (the experiments demand large quantity of computations). The amount demanded for powerful personal computers fits with their international prices.

I period (1-6 months) II period (7-12 months) III period (13-18 months) IV period (19-24 months)

Interim outcomes/List of countable indicators

Interim outcomes/List of countable indicators

Interim outcomes/List of countable indicators

Interim outcomes/List of countable indicators

1. Organizing a research seminar

Preparation of one technical report and one article Organizing a research seminar

One article. Software demonstration, Preparation of the documentation.

References

[Ant06] J. Antidze. On One Approach to Automatic Semantic Analysis of Phrases of a Natural Language. Bulletin of The Georgian Academy of Sciences, vol.174, 1, 2006.

[Ant07] J. Antidze. Software Tools for Morphological and Syntactic Analysis of Natural Language Texts. Internet Academy, Georgian Electronic Scientific Journals: Computer Sciences and Telecommunications, 1(12), 2007, IISN 1512-1232.

[BGK03] P. Buneman, M. Grohe, and C. Koch. Path queries on compressed XML. In Proc. VLDB 2003, pages 141–152. Morgan Kaufmann, 2003.

[BLM08] G. Busatto, M. Lohrey, and S. Maneth. 2008. Efficient memory representation of XML document trees. Information Systems 33, 4–5, 456–474.

[CD96] Codognet, P., Diaz, D., 1996. Compiling Constraints in CLP(FD). J. Log. Program. 27(3): 185-226.

[CDF+10] Coelho, J., Dundua, B., Florido, B., Kutsia, T., 2010. A Rule-Based Approach to XML Processing and Web Reasoning. In: P. Hitzler and T. Lukasiewicz, editors, Proc. 4th International Conference on Web Reasoning and Rule Systems, RR 2010. Volume 6333 of LNCS. Springer, 164-172.

[DFK+14] B. Dundua, M. Florido, T. Kutsia and M. Marin. Constraint Logic Programming for Hedges:

a Semantic Reconstruction. In: M. Codish and E. Sumii, editors. Proceedings of the 12th International Symposium on Functional and Logic Programming, FLOPS 2014. Volume 8475 of Lecture Notes in Computer Science. Springer, 2014. 285{301. To appear.

[DK10] B. Dundua and T. Kutsia. PρLog, version 0.9, 2010.
http://www.risc.jku.at/people/tkutsia/software.html.
[DKM09] Dundua, B., Kutsia, T., Marin, M., 2009. Strategies in PρLog. In: M. Fernandez, editor, 9th International Workshop on Reduction Strategies in Rewriting and Programming, WRS 2009.
Electronic Proceedings in Theoretical Computer Science 15, 2010, 32-43.
[GGS+08] A. Gascón, G. Godoy, and M. Schmidt-Schauß. Context matching for compressed terms. In 23rd Annual IEEE Symposium on Logic in Computer Science (LICS 2008), pages 93-102. IEEE Computer Society, 2008.
[GGS+09] A. Gascón, G. Godoy, and M. Schmidt-Schauß. Unification with singleton tree grammars. In 20. RTA 2009, volume 5595 of LNCS, pages 365-379. Springer, 2009.
[GT07] Sumit Gulwani and Ashish Tiwari. Computing procedure summaries for interprocedural analysis. In Rocco De Nicola, editor, ESOP, volume 4421 of Lecture Notes in Computer Science, pages 253{267. Springer, 2007.
[Ham97] M. Hamana Term rewriting with sequences. In: Proc. of the First International Theorema Workshop. Technical report 97–20, RISC. Johannes Kepler University, Linz, Austria
[Hon93] Hong, H., 1993. RISC-CLP(Real): Logic Programming with Non-linear Constraints over the Reals, in: Constraint Logic Programming: Selected Research, MIT Press, 133-159.
[Kol98]Alexander Koller. Evaluating context unication for semantic underspecification. In Proceedings of the Third ESSLLI Student Session, pages 188-199, 1998.
[kut02] Temur Kutsia. Theorem proving with sequence variables and flexible arity symbols. In Matthias Baaz and Andrei Voronkov, editors, LPAR, volume 2514 of Lecture Notes in Computer Science, pages 278-291. Springer, 2002.
[LB10] Catherine Lai, Steven Bird: Querying Linguistic Trees. Journal of Logic, Language and Information 19(1): 53-73 (2010)
[Lyo87] J. Lloyd. Foundations of Logic Programming. Springer-Verlag, 2nd edition, 1987. 5.1
[Men11] Christopher Menzel. Knowledge representation, the world wide web, and the evolution of logic. Synthese, 182(2):269-295, 2011.
[MK06] Marin, M., Kutsia, T., 2006. Foundations of the Rule-Based System RhoLog. J. Applied Non- Classical Logics, 16(1-2):151-168.

[Moh96] Markus Mohnen. Context patterns in haskell. In Werner E. Kluge, editor, Implementation of Functional Languages, volume 1268 of Lecture Notes in Computer Science, pages 41{57. Springer, 1996.
[JM94] Jaffar, J., Maher, M.J., 1994. Constraint Logic Programming: A Survey. J. Log. Program. 19/20: 503-581.
[JMM+98] Jaffar, J., Maher, M.J., Marriott, K., Stuckey, P.J. 1998. The Semantics of Constraint Logic Programs. J. Log. Program. 37(1-3): 1-46.
[JMP+92] Jaffar, J., Michaylov, S., Stuckey, P., Yap, R, 1992. The CLP(R) Language and System, ACM Transactions on Programming Languages, 14(3), 339-395.
[Pau] Paun, G. Introduction to membrane computing, In: Perez-Jemenez, M.J. et al.(Eds) Applications of Membrane Computing. Springer, Heidelberg, 2006, pp. 1-42.
[Wol03] S. Wolfram. The Mathematica Book. Wolfram Media Inc., 5th edition, 2003.

3. Key Personnel

1. Antidze Jemal, Principal Investigator, Doctor, 1935-06-25
2. Dundua Besik, Researcher, Master, 1980-03-22
3. Kardava Irakli, Researcher, Master, 1986-11-01

Project Key Personal's Curriculum Vitaes (CV)

Name, Surname: Jemal Antidze

Education

1. 1978, Tbilisi State University, Software of computers and systems, Senior scientist
2. 1966, Tbilisi State University, Defend of doctoral thesis, Doctor of physics and mathematics
3. 1953-58, Tbilisi State University, Mathematics, Teacher

Relevant work experience

1. 2011 Director Scientific Research Institute of Mathematics and Information Technology Sokhumi State University
2. 2010 Scientific Researcher Institute of Applied Mathematics Tbilisi State University
3. 2006-09 Associated Professor Institute of Computer Sciences Tbilisi State University
4. 1973-2006 Head of Department Institute of Applied Mathematics Tbilisi State University
5. From 1913 Sokhumi State University

Number of Publications: 97

List of publications in last 10 years (Up to 10 for Principal Investigator, up to 5 for all other participants)

1. 2013, The Software for Composition of Some Natural Language Words Lecture Notes on Software Engineering, vol.1, #3, pages 295-297, http://www.lnse.org J. Antidze, N. Gulua, I. Kardava
2. 2012, Software Tools for Some Natural Language Texts Computer Prossecing, Computer Technology and Application,vol.3, number 3,219-225pages, impact factor=3, J.Antidze, N.Gulua
3. 2011, Generalized Tools for Computer Realization of Natural Language Models, Proceedings of "The Eleventh International Conference on Pattern Recognition and Information Processing", pages 362-365, Minsk, Belarus J.Antidze, N.Gulua.
4. 2010, Software for Processing of Natural Language Texts Proceedings of Third International Conference "Problems of Cybernetics and Informatics", Volume1, pages 114-117, http://fpv.science.tsu.ge/baku.mht J.Antidze, N.Gulua
5. 2010, On Complete Computer Morphological and Syntactic Analysis of Georgian Texts, Proceedings of the Seventh International Conference Internet–Education – Science, vol. 1(11), pages 214-217, http://fpv.science.tsu.ge/vinitsa1.mht J.Antidze, N.Gulua

6. 2010, Software Tools for Computer Realization of Morphological and Syntactic Models of Georgian Texts, International Scientific Journal Optoelectronic Information-Power Technologies,#2(1920), pages 92-97, J.Antidze, N.Gulua
7. 2010, On Complete Machine Translation of Texts from Georgian Language Computer Sciences and Telecommunications,#1(24), pages 54-63, http://gesj.internet academy.org.ge, J.Antidze, N.Gulua

Participation in research grants projects (not exceeding 5)

1. 2013-2014, Head of the project A Pattern Calculus with Hedge Variables, Rustaveli National Scientific Foundation
2. 2010, Head of Project Rule based programming with second order and sequential variables, Shota Rustaveli National Science Foundation

Participation in international scientific forums (not exceeding 5)

1. 2013, International Conference on Software and Computer Applications, Software Tools for Computer Modeling of a Natural Language Texts, Keynote Speech Paris, France
2. 2010, Third International Conference "Problem s of Cybernetics and Informatics", Software for Processing of Natural Language Texts Baku, Azerbaijan
3. 2010, The Seventh International Conference Internet – Education – Science, On Complete Computer Morphological and Syntactic Analysis of Georgian Texts, Plenary Session, Vynnitsia, Ukraine
4. 2008, Symposium – Natural Language Processing, Georgian Language and Computer Technologies, On Full Morphological, Syntactic and Partly Semantic Analysis of Georgian Language and Machine Translation from Georgian Language, Institute of Linguistics of Georgian Academy of Sciences, Tbilisi
5. 2008, International Conference in Artificial Intelligence and Symbolic Computation(AISC), Special Language with Constraints, its Realization and Application for Morphological and Syntactic Analysis of Georgian Texts, Birmingham University, 30 July-01 August

Project Key Personal's Curriculum Vitaes (CV)

Name, Surname: Besik Dundua

Education

1. 2001-2003, Ivane Javakhishvili Tbilisi State University, Mathematics, Master of mathematics
2. 1997-2001, Ivane Javakhishvili Tbilisi State University, Mathematics, Bachelor of Mathematics

Relevant work experience

1. 2006, Researcher Tbilisi State University, Vekua Institute of Applied Mathematics.
2. 2004-2005, Guest researcher, Johannes Kepler University, Linz, Austria. Research Institute for Symbolic Computation

Number of Publications: 12

List of publications in last 10 years (Up to 10 for Principal Investigator, up to 5 for all other participants)

1. 2014, A Semantic Reconstruction. In: M. Codish and E. Sumii, editors. In: M. Codish and E. Sumii, editors. Proceedings of the 12th International Symposium on Functional and Logic Programming, FLOPS 2014.Volume 8475 of Lecture Notes in Computer Science. Springer, 2014. 285-301. B. Dundua, M. Florido, T. Kutsia and M. Marin.
2. 2013, A Confluent Pattern Calculus with Hedge Variables, N. Hirokawa and V. van O ostrom , editors, 2nd International Workshop on Confluence, IWC '13. June 28, 2013, 41-45, Eindhoven, The Netherlands, S. Alves, B. Dundua, M. Florido, and T. Kutsia
3. 2010, Strategies in PρLog. Electronic Proceedings in Theoretical Computer Science, pp. 32-43. 2010. ISSN 2075-2180 . Besik Dundua, Temur Kutsia, Mircea Marin.

4. 2010, Trust and Belief, Interrelation. Fernandez, Adriana Giret and Vicente Julian, editors, Proceedings of the Third Workshop on Agreement Technologies, WAT 2010, Bahia Blanca, Argentina, volume 657 of CEUR Workshop Proceedings, pages 35-42, ISSN 1613-0073. Besik Dundua and Levan Uridia.
5. 2010, A Rule-Based Approach to XML Processing and Web Reasoning Lecture Notes in Computer Science 6333, pp. 164-172. Springer, ISSN 0302-9743, ISBN 978-3-642-15917-6. Jorge Coelho, Besik Dundua, Mario Florido and Temur Kutsia.

Participation in research grants projects (not exceeding 5)

1. 2012-2015, Researcher, Pattern Calculus with Sequence Variables, Shota Rustaveli National Science Foundation, Second Order Programming with Georgian National
2. 2010, Head of Project, Transformational Rules with Sequence Variables, Georgian National Science Foundation
3. 2010, Researcher Rule-Based Programming with Second-Order and Sequence Variables, Georgian National Science Foundation

Project Key Personal's Curriculum Vitaes (CV)

Name, Surname: Irakli Kardava

Education

1. 2013, Sokhum i State University Mathematical Modeling and Information Technology, PhD Student
2. 2010-2012, Sokhumi state university information technology master

Relevant work experience

1. 2010-2012, Scientist, Ilia Vekua Institute of Applied Mathematics(VIAM) of Ivane Javakhishvili Tbilisi State University (TSU)

Nubmer of Publications: 4

List of publications in last 10 years (Up to 10 for Principal Investigator, up to 5 for all other participants)

1. 2013, The Software for Composition of Some Natural Languages' Words, Lecture Notes on Software Engineering, vol.1, #3, August, Singapure, pages 295-297. J.Antidze, N.Gulua, I.Kardava
2. 2012, The Software for Composing Georgian Words, Transactions of The International Scientific Conference Dedicated to the 90 Anniversary of Georgian Technical University, Pages 367-371, Tbilisi, Georgia, J.Antidze, N.Gulua, I.Kardava

Participation in research grants projects (not exceeding 5)

1. 2013-2014, Researcher Scientist, Rule based programming with second order and sequential variables, Rustaveli National Scientific Foundation