数式処理 *J.JSSAC (2003)*
Vol. 10, No. 1, pp. 23 - 33
特集 "Quantifier Elimination"

REDLOG – First-Order Logic for the Masses

Andreas DOLZMANN* Andreas SEIDL†
University of Passau, Germany University of Passau, Germany

Abstract

REDLOG is a system for computing with first-order logic and propositional logic with quantification. It is tightly integrated into the computer algebra system REDUCE. Assuming a broad audience, we motivate the use of first-order logic to model problems. Then, by employing quantifier elimination methods, simplification techniques and normal form computations, highly non-trivial problems can be solved. We give an overview of REDLOG's capabilities, features and applications. Finally we explain how computer algebra benefits from computer logic and vice versa.

1 Introduction

Propositional logic is a well-established tool in mathematics, computer science and engineering. In short, Boolean expressions are built from Boolean variables and Boolean connectives, which include $\wedge$,$\vee$, $\neg$ and $\longrightarrow$. If truth values for all Boolean variables are given, a Boolean expression can be evaluated to either true or false.

First-order logic generalizes this approach in two ways. First, the Boolean variables are replaced by more sophisticated atomic formulas, which in turn can contain variables. Admissible atomic formulas are given by fixing a *language*. Second, by allowing variables to be existentially or universally quantified, the expressiveness of such formulas is drastically increased.

The following would be a first-order formula over a language, which includes a constant 0 and an ordering $<$, such as the language of ordered rings:

$$\varphi_{\text{minpos}} := \exists x\big(x > 0 \wedge \forall y\big(y > 0 \longrightarrow (y > x \vee y = x)\big)\big).$$

Such formulas have a more complex structure than expressions from propositional logic, but on the other hand much more interesting things can be described. They are still sufficiently comprehensible by humans.

*http://www.fmi.uni-passau.de/~dolzmann
†http://www.fmi.uni-passau.de/~seidl

© 2003 Japan Society for Symbolic and Algebraic Computation

To attach meaning to φ_{minpos}, we first have to fix a *structure*, i.e. a domain and semantics for 0 and $<$. First, let us consider the reals $\mathbb{R}$ and interpret 0 and $<$ as usual. Then φ_{minpos} states the following:

> "There exists an x such that x is positive, and for all y such that y is positive as well, y is greater than or equal to x."

In other words, φ_{minpos} holds in $\mathbb{R}$ if and only if there exists a minimal positive element. As an easy proof shows that such an element does not exist, φ_{minpos} is equivalent to false in $\mathbb{R}$. Second, consider the natural numbers $\mathbb{N}$ and interpret 0 and $<$ as usual. Now φ_{minpos} is certainly equivalent to true, as 1 is a minimal positive element. Altogether, by fixing a structure and by using our understanding of the specific settings, we were able to find for a given formula a simpler, quantifier-free one. Note that this example was a special case, as all occurring variables lie within the scope of a quantifier. In general, *free* variables of the formula can occur in the quantifier-free description as well. We call free variables also *parameters* in this context. Consider the formula $\exists x(x^2+bx+c=0)$. A simpler condition on b and c, such that the quadratic equation has a real root, is $b^2-4c \geq 0$. Finding simpler formulas is what quantifier elimination is all about.

More precisely, we define: For a given language $\mathcal{L}$ and an $\mathcal{L}$-structure $\mathbf{A}$, $\mathbf{A}$ admits *quantifier elimination* over $\mathcal{L}$ if and only if for any formula φ there exists a quantifier-free formula φ' such that every free variable of φ' is a free variable in φ and $\varphi \longleftrightarrow \varphi'$ holds in $\mathbf{A}$. Similarly quantifier elimination wrt. a class of structures over a fixed language is defined.

The question arises, for which classes of structures over which languages quantifier elimination is algorithmically possible. There are striking positive and negative results. The class of real closed fields, which includes the reals $\mathbb{R}$, over the language of ordered rings admits quantifier elimination. Integers over the language of rings do not. If one, however, restricts the language to Presburger arithmetic, they do.

The positive results, numerous possible applications and the fact that quantifier elimination algorithms are too complicated to be executed by hand, except for very simple examples, raise the desire to turn the theoretical possibility into a practical implementation. This is where REDLOG comes into play. The development of REDLOG was started in 1992 by the first author together with T. Sturm. Meanwhile the second author has joined the development team.

The plan of this paper is as follows: In the next section we show an example application. We learn that it is an advantage to have several quantifier elimination methods and further simplification and normal form computation routines at hand. Then in Section 3 we present the REDLOG system. Quantifier elimination, e.g. for the reals, has many important applications: In Section 4 we give an overview. We finally make the point, that integrating a logic system into a computer algebra system is of mutual benefit and avoids

isolated solutions.

2 An Illustrating Example

In this section we see a classical example from real algebraic geometry. The example comes in two parts. In the first part, we solve a problem by simply applying quantifier elimination. The way the subsequent problem in the second part is treated, however, communicates a more subtle point: If a problem is too large to be automatically solved straightforwardly by quantifier elimination, there are still other options to be considered, which can lead to success. All times given are on a 2 GHz Pentium 4 using up to 128 MB of memory.

In this section, the reals and the language of ordered rings is the setting we operate in. Here quantifier elimination is possible and implemented in the context OFSF of REDLOG. There are more contexts available, as will be explained in Section 3. Interesting cutting-edge applications are by far not limited to real algebraic geometry, as will be seen in the overview given in Section 4.

2.1 Membership of the Real Enneper Surface

The real Enneper surface, see e.g. [5], is defined parametrically by

$$x = 3u + 3uv^2 - u^3 \quad y = 3v + 3u^2v - v^3 \quad z = 3u^2 - 3v^2,$$

in other words as the image of the function

$$f : \mathbb{R}^2 \longrightarrow \mathbb{R}^3 \quad \text{with} \quad f(u,v) = (3u + 3uv^2 - u^3, 3v + 3u^2v - v^3, 3u^2 - 3v^2).$$

The complex counterpart of the above surface dates back to the German mathematician Alfred Enneper who constructed the surface in 1863. It is a well known minimal surface.

Given a point $(x, y, z) \in \mathbb{R}^3$ we cannot easily decide, whether it is contained in the Enneper surface or not. This task is necessary e.g. for plotting the surface. A point is contained in the surface, if there is a $(u, v) \in \mathbb{R}^2$, such that $f(u, v) = (x, y, z)$. This condition is given by the following formula:

$$\varphi := \exists u \exists v \big(x = 3u + 3uv^2 - u^3 \wedge y = 3v + 3u^2v - v^3 \wedge z = 3u^2 - 3v^2 \big).$$

We are looking for a formula ψ in the variables x, y, and z such that $\psi(x, y, z)$ is true if and only if $\varphi(x, y, z)$ is true. In other words, we want to find a quantifier-free equivalent for φ. By real quantifier elimination, such a formula can be found algorithmically.

There are three different methods for quantifier elimination available for this setting: The virtual substitution method, a method based on partial cylindrical algebraic decom-

position (CAD) and a method based on multivariate real root counting. Each method has advantages and disadvantages, as explained in Section 3.

The third method allows to compute in about 2.6 s a quantifier-free equivalent ψ containing 164 atomic formulas. Both other methods fail to eliminate the formula, because of degree restrictions in the first case, and limited resources for the CAD method. The computed result allows to decide in less than 0.1 s whether a given point belongs to the Enneper surface.

2.2 Real and Complex description of the Enneper Surface

We have computed a description ψ of the real Enneper surface containing no quantifiers. For the complex Enneper surface, defined as the real one, but for complex variables, the following description is known:

$$\begin{aligned}\psi'' \quad := \quad & 19683x^6 - 59049x^4y^2 + 10935x^4z^3 + 118098x^4z^2 - 59049x^4z + \\ & 59049x^2y^4 + 56862x^2y^2z^3 + 118098x^2y^2z + 1296x^2z^6 + \\ & 34992x^2z^5 + 174960x^2z^4 - 314928x^2z^3 - 19683y^6 + 10935y^4z^3 - \\ & 118098y^4z^2 - 59049y^4z - 1296y^2z^6 + 34992y^2z^5 - 174960y^2z^4 - \\ & 314928y^2z^3 - 64z^9 + 10368z^7 - 419904z^5 = 0.\end{aligned}$$

The most natural way to find out whether φ'' is a valid description in the reals, too, would be to apply quantifier elimination to

$$\forall x \forall y \forall z (\psi \longleftrightarrow \psi'').$$

This problem, however, turns out to be too large to be practically feasible. But we still have not run out of options.

REDLOG provides several strategies for simplifying formulas [11]. By applying an advanced simplifier, which uses Gröbner bases internally, we can compute in 1.6 s a disjunctive normal form ψ' of our result. It contains 11 conjunctions with 57 atomic formulas in total. Let us thus write ψ' as $\psi_1 \vee \cdots \vee \psi_{11}$ with conjunctions ψ_i. Unfortunately, even with the smaller but equivalent version ψ' used for ψ, the formula above still cannot be eliminated within reasonable time.

Besides simplification of a formula, REDLOG can simplify a formula wrt. a given *theory*, i.e. a set of atomic formulas considered as a conjunction. Then the simplification result is equivalent to the input formula for all values, for which the theory holds. For the computed disjunctive normal form we can simplify the equation ψ'' wrt. each conjunction ψ_i, viewed as the set of the contained atomic formulas. REDLOG computes in each case `true` within 0.1 s. So far we have proved $\psi' \longrightarrow \psi''$.

It still remains to prove the other direction $\psi'' \longrightarrow (\psi_1 \vee \ldots \vee \psi_{11})$. Again, we break this problem down into smaller ones by considering the four cases $x = 0$, $y = 0$, $z = 0$ and $x \neq 0 \wedge y \neq 0 \wedge z \neq 0$. The first three cases take less than 0.2 s. As for the forth case, by finding

$$\exists x \exists y \exists z (x \neq 0 \wedge y \neq 0 \wedge z \neq 0 \wedge \psi_i)$$

to be false except for $i = 1$ and $i = 3$, it suffices to prove

$$(x \neq 0 \wedge y \neq 0 \wedge z \neq 0) \longrightarrow \big(\psi'' \longrightarrow (\psi_1 \vee \psi_3)\big).$$

By making further case distinctions, and by further splitting up these cases, we finally can finish the proof. However, at this point we can safely end the example, as we have seen quantifier elimination, simplification and normal form computation techniques in action. The overall computing time amounts to about 8 s. In 1998 the first author succeeded to prove the equivalence between ψ and ψ'' by using REDLOG [7]. At that time this was an open problem. The overall computation time in 1998 was 30 min on a 140 MHz Sun Ultra Sparc 1 workstation.

3 REDLOG, the REDUCE Logic System

REDLOG extends the REDUCE computer algebra system to a computer logic system. More precisely, REDLOG's domain is first-order logic. Recent work was done to make propositional calculus with quantification of Boolean variables available in this framework [19]. As most prominent feature, REDLOG offers quantifier elimination. As explained in the introduction, it is necessary to fix a language and a class of structures over this language. In REDLOG's terms, this is called a *context*. Fixing a context is not only imposed by the wish to eliminate quantifiers. Advanced simplification techniques as well as sophisticated normal form computation methods depend on a context-specific part. The impact of this on design issues of REDLOG is discussed in greater detail in [10].

Next we give an overview of the currently available contexts. Each context includes procedures for Boolean normal form computation, i.e. conjunctive and disjunctive normal form computation. In addition, minimal prenex normal forms, anti-prenex normal forms and negation normal forms can be computed in each context.

- OFSF: real closed fields over the language of ordered rings. This class includes the reals $\mathbb{R}$. There are three quantifier elimination methods available.

  Originally, linear quantifier elimination based on the *virtual substitution* method [25] was implemented. This was then extended to deal with higher degrees [26] in special cases. For problems with lots of parameters, many variables and low degrees this method is the best choice, as parameters do not contribute to the complexity in a relevant way. The output formulas may contain redundancies.

The second method, based on partial *cylindrical algebraic decomposition* (CAD) [2, 3], has no degree restrictions. However, as parameters contribute fully to the complexity, this method is preferable for problems with few variables. It produces very concise output formulas.

The third method is based on *real root counting* [27, 6]. It is preferable in case a formula has a simple Boolean structure and contains mainly equations.

Variants of the quantifier elimination procedures include *extended* quantifier elimination, which computes not only a quantifier-free equivalent but sample points, *generic* and *local* quantifier elimination [12, 14].

Besides quantifier elimination there are several simplification techniques implemented. They include a powerful standard simplifier for arbitrary formulas and a simplifier, which automatically generates case distinctions. Furthermore, there are context-specific improvements for Boolean normal form computation.

- ACFSF: algebraically closed fields over the language of fields. This includes the complex numbers $\mathbb{C}$. Quantifier elimination based on comprehensive Gröbner bases, including a generic variant, is provided in this context. All simplification methods and normal form computations described for OFSF are also available for this context.

- DVFSF: discretely valued fields over a one sorted language with abstract divisibility. This includes the p-adics. This context provides a linear quantifier elimination and an adapted standard simplifier.

- PASF, a context for Presburger arithmetic, i.e. for computing with linear formulas over the integers $\mathbb{Z}$. This context is currently still in preparation.

- IBALP: initial Boolean algebras over the language of Boolean algebras. All members of this class are isomorphic to the two-element Boolean algebra. This context provides propositional logic with quantification [19], and includes also quantifier elimination, an adapted standard simplifier and context-specific improvements for Boolean normal form computation.

We recall the observation made in the second part of the example in Section 2. There we have seen, that not by simply applying quantifier elimination alone, but by combining it with simplification and normal form computation techniques, we reached success. This is an often underestimated issue: successful quantifier elimination strongly depends on powerful simplification and normal form computation techniques.

Last but not least, next to the elaborate methods it should be mentioned that also numerous tools on formulas and for constructing them are available. These tools include for-loops to generate large conjunctions and disjunctions, syntactic substitution, dropping

prenex quantifiers, universal and existential closure, multiplicity list of terms and atomic formulas, list of free or bound variables and number of atomic formulas.

Even though algorithms may be fully or partly context-dependent, REDLOG provides a uniform interface: the canonical procedure call is the same for all contexts. As for quantifier elimination, some contexts implement several methods. In this case we decide heuristically, which algorithms to use, depending on the input formula. In addition there are front-ends for all methods available. Furthermore there are switches to turn features on or off, or to decide between complementary strategies.

REDLOG differs significantly from theorem proving systems, which usually have uniform algorithms, no knowledge about the language, and all properties of the structure given explicitly by axioms. REDLOG's algorithms, in contrast, depend fully or in part on the special properties of the structures they were designed for. As an example, a theorem proving system may derive the idempotent laws for Boolean algebras from given axioms regarding the existence of identities, the distributive laws and the existence of complement. REDLOG, however, when switched to the context IBALP, proves all of these laws without needing further information. A link between the two approaches is given by constraint logic programming [4, 24].

We refer the reader to the REDLOG user manual, which can be downloaded from the REDLOG page[1)]. On this page there is a link to REMIS, our online example database.

4 Applications Overview

In this section we give an overview of applications and examples of quantifier elimination, which lie within the scope of REDLOG.

The context OFSF is REDLOG's oldest and most elaborate one, so it is not astonishing that most applications use real quantifier elimination.

- The analysis of partial differential equations is one of the best studied application area. This includes stability analysis [15] of PDE's and a method for deciding them to be elliptic [20].
- Applications of quantifier elimination in control theory [1, 17] and in the analysis of hybrid systems, in particular for the computation of the reachability space, are also extensively studied [1, 17].
- Ioakimidies has applied REDLOG in the area of theoretical mechanics [16].
- Quantifier elimination and simplification methods of REDLOG can also be applied in the area of Hopf bifurcations [18].

[1)] http://www.fmi.uni-passau.de/~redlog

- Motion planning for one or more robots in a time dependent environment was also studied [29, 28, 13].
- Sturm suggested to use REDLOG for the design, analysis and diagnosis of electrical networks [22].
- The first author has described in his doctoral dissertation how to solve scheduling problems for the traditional dedicated machine model and for project networks by applying the extended quantifier elimination of REDLOG [8].
- Problems in the area of geometry and computer aided design may be also solved by quantifier elimination. [23, 21].

Quantifier elimination for the p-adics within the context DVFSF was used to implement an algorithm for computing integer solutions of systems of congruence wrt. prime power moduli.

Application examples in digital circuit design and testing of faulted circuits are given by Sturm and the second author in [19] by using the newly available context IBALP.

The REDUCE package for computing comprehensive Gröbner bases uses the algebraically closed field context ACFSF for simplifying and testing conditions on the parameters.

5 Computer Logic meets Computer Algebra

As seen in the second part of the example in Section 2 there might not necessarily be a straightforward one-step solution of a problem. To find a solution, getting interactively involved with the system is required. Such a course of action is greatly facilitated by REDLOG's seamless integration into a computer algebra system. There are further, less obvious benefits from integrating a logic system to a computer algebra system than the ease of use.

Consider the expression a^2/a. A computer algebra system usually simplifies this to a. If a is considered to be a transcendental element, the simplification is in fact valid. However, if a is considered to be a variable, which can have arbitrary numbers from a domain as values, this simplification is wrong, because division is a partial function and division by $a = 0$ is not defined.

Guarded expressions [9] can deal with partially defined functions as well as piecewise defined functions. The key idea is to assign to each partially defined expression a *guard*, i.e. a quantifier-free first-order formula, which states under which conditions the expression is valid. This idea generalizes to piecewise defined functions by considering sets of guarded expressions. The functionality of REDLOG allows to efficiently deal with the guards, simplifying them or testing them to be a contradiction or a tautology. Furthermore, this concept allows to systematically simplify algebraic expressions by using logical knowledge of the

expressions and its subexpressions. As an example, consider the possible simplification of $\text{sign}(x) \cdot x - |x|$ to 0, as shown in the above-mentioned paper.

The application of guards is rewarding in particular for all parametric non-uniform computations. For example, REDUCE uses this concept for computing comprehensive Gröbner bases, i.e. the parametric variant of Gröbner bases. In this package guards are handled by using REDLOG.

6 Conclusions

REDLOG is a package that extends the REDUCE computer algebra system to a *computer logic* system. The integration of computer logic and computer algebra turns out to be of *mutual* benefit. REDLOG's domain is first-order logic, where a *context*, i.e. a language and a class of structures over this language, is fixed. REDLOG provides efficiently implemented *quantifier elimination* methods , offers powerful *simplification* and *normal form* computation techniques. It comes with numerous *tools* for constructing and processing formulas and is easily *available* as part of the computer algebra system REDUCE. REDLOG has been around now for 10 years and has been successfully used in various applications, including theoretical mechanics, stability problems, control theory, design and analysis of electrical networks and digital circuits, scheduling, geometrical theorem proving, computer aided design, partial differential equations and real algebraic geometry.

References

[1] Chaouki T. Abdallah, Peter Dorato, Wei Yang, Richard Liska, and Stanly Steinberg. Applications of quantifier elimination theory to control system design. In *Proceedings of the 4th IEEE Mediterranean Symposium on Control and Automation*, pages 340–345. IEEE, 1996.

[2] George E. Collins. Quantifier elimination for the elementary theory of real closed fields by cylindrical algebraic decomposition. In H. Brakhage, editor, *Automata Theory and Formal Languages. 2nd GI Conference*, volume 33 of *Lecture Notes in Computer Science*, pages 134–183. Springer-Verlag, Berlin, Heidelberg, New York, 1975.

[3] George E. Collins and Hoon Hong. Partial cylindrical algebraic decomposition for quantifier elimination. *Journal of Symbolic Computation*, 12(3):299–328, September 1991.

[4] Alain Colmerauer. Prolog III. *Communications of the ACM*, 33(7):70–90, July 1990.

[5] David Cox, John Little, and Donald O'Shea. *Ideals, Varieties and Algorithms.* Un-

dergraduate Texts in Mathematics. Springer-Verlag, New York, Berlin, Heidelberg, 1992.

[6] Andreas Dolzmann. Reelle Quantorenelimination durch parametrisches Zählen von Nullstellen. Diploma thesis, Universität Passau, D-94030 Passau, Germany, November 1994.

[7] Andreas Dolzmann. Solving geometric problems with real quantifier elimination. In Xiao-Shan Gao, Dongming Wang, and Lu Yang, editors, *Automated Deduction in Geometry*, volume 1669 of *Lecture Notes in Artificial Intelligence (Subseries of LNCS)*, pages 14–29. Springer-Verlag, Berlin Heidelberg, 1999.

[8] Andreas Dolzmann. *Algorithmic Strategies for Applicable Real Quantifier Elimination*. Doctoral dissertation, Department of Mathematics and Computer Science. University of Passau, Germany, D-94030 Passau, Germany, March 2000.

[9] Andreas Dolzmann and Thomas Sturm. Guarded expressions in practice. In Wolfgang W. Küchlin, editor, *Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation (ISSAC 97)*, pages 376–383, Maui, HI, July 1997. ACM, ACM Press, New York, 1997.

[10] Andreas Dolzmann and Thomas Sturm. Redlog: Computer algebra meets computer logic. *ACM SIGSAM Bulletin*, 31(2):2–9, June 1997.

[11] Andreas Dolzmann and Thomas Sturm. Simplification of quantifier-free formulae over ordered fields. *Journal of Symbolic Computation*, 24(2):209–231, August 1997.

[12] Andreas Dolzmann, Thomas Sturm, and Volker Weispfenning. A new approach for automatic theorem proving in real geometry. *Journal of Automated Reasoning*, 21(3):357–380, 1998.

[13] Andreas Dolzmann and Volker Weispfenning. Multiple object semilinear motion planning. To appear.

[14] Andreas Dolzmann and Volker Weispfenning. Local quantifier elimination. In Carlo Traverso, editor, *Proceedings of the 2000 International Symposium on Symbolic and Algebraic Computation (ISSAC 2000), St Andrews, Scotland*, pages 86–94. ACM Press, New York, NY, August 2000.

[15] Hoon Hong, Richard Liska, and Stanly Steinberg. Testing stability by quantifier elimination. *Journal of Symbolic Computation*, 24(2):161–187, August 1997. Special issue on applications of quantifier elimination.

[16] Nikolaos I. Ioakimidis. REDLOG-aided derivation of feasibility conditions in applied mechanics and engineering problems under simple inequality constraints. *Journal of Mechanical Engineering (Strojnícky Časopis)*, 50(1):58–69, 1999.

[17] Mats Jirstrand. Nonlinear control system design by quantifier elimination. *Journal of Symbolic Computation*, 24(2):137–152, August 1997. Special issue on applications of quantifier elimination.

[18] M'hammed el Kahoui and Andreas Weber. Deciding hopf bifurcations by quantifier elimination in a software component architecture. *Journal of Symbolic Computation*, 30(2):137–149, August 2000.

[19] Andreas Seidl and Thomas Sturm. Boolean quantification in a first-order context. Technical report, FMI, Universität Passau, 2003.

[20] Werner M. Seiler and Andreas Weber. Deciding ellipticity by quantifier elimination. To appear.

[21] Thomas Sturm. *Real Quantifier Elimination in Geometry.* Doctoral dissertation, Department of Mathematics and Computer Science. University of Passau, Germany, D-94030 Passau, Germany, December 1999.

[22] Thomas Sturm. Reasoning over networks by symbolic methods. *Applicable Algebra in Engineering, Communication and Computing*, 10(1):79–96, September 1999.

[23] Thomas Sturm. An algebraic approach to offsetting and blending of solids. In V. G. Ganzha, E. W. Mayr, and E. V. Vorozhtsov, editors, *Computer Algebra in Scientific Computing. Proceedings of the CASC 2000*, pages 367–382. Springer, Berlin, 2000.

[24] Thomas Sturm. Quantifier elimination-based constraint logic programming. Technical Report MIP-0202, FMI, Universität Passau, D-94030 Passau, Germany, January 2002.

[25] Volker Weispfenning. The complexity of linear problems in fields. *Journal of Symbolic Computation*, 5(1&2):3–27, February–April 1988.

[26] Volker Weispfenning. Quantifier elimination for real algebra—the quadratic case and beyond. *Applicable Algebra in Engineering Communication and Computing*, 8(2):85–101, February 1997.

[27] Volker Weispfenning. A new approach to quantifier elimination for real algebra. In B.F. Caviness and J.R. Johnson, editors, *Quantifier Elimination and Cylindrical Algebraic Decomposition*, Texts and Monographs in Symbolic Computation, pages 376–392. Springer, Wien, New York, 1998.

[28] Volker Weispfenning. Semilinear motion planning among moving objects in REDLOG. In V. G. Ganzha and E. W. Mayr, editors, *Computer Algebra in Scientific Computing. Proceedings of the CASC 2001*, pages 541–553. Springer, Berlin, 2001.

[29] Volker Weispfenning. Semilinear motion planning in REDLOG. *Applicable Algebra in Engineering, Communication and Computing*, 12:455–475, June 2001.