Automated Theorem Proving in Loop Theory

J. D. Phillips
Wabash College, USA

David Stanovský [*]
Charles University, Czech Republic

Abstract

In this paper we compare the performance of various automated theorem provers on nearly all of the theorems in loop theory known to have been obtained with the assistance of automated theorem provers. Our analysis yields some surprising results, e.g., the theorem prover most often used by loop theorists doesn't necessarily yield the best performance.

1 Introduction

Automated reasoning tools have had great impact on loop theory over the past decade, both in finding proofs and in constructing examples. It is widely believed that these achievements have transformed loop theory, both as a collection of deep results, as well as the mode of inquiry itself. Automated reasoning tools are now standard in loop theory.

To date, all automated proofs in loop theory have been obtained by Prover9 [McC05] or its predecessor Otter [McC03], and all models have been generated by SEM [ZZ] , Mace4 [McC05], and recently the Loops package for GAP [NV] which G. Nagy used to help find the first nonMoufang, finite simple Bol loop (definitions to follow in section 2), thus solving one of the oldest open problems in loop theory. The present paper is devoted to automated theorem proving. For model building, it seems that GAP/Loops is far better than general purpose automated reasoning tools, especially in certain of the more well known varieties of loops, as it exploits the underlying group theory with its fast algorithms. (Also, most interesting problems about finding finite loops either include properties that cannot be easily formalized in first order theory (such as simplicity), or are known to have lower bound at least several hundred elements.)

While [Phi03] is an introduction to automated reasoning for loop theorists, the present paper is intended as its complement: for computer scientists as an introduction to one of the areas in algebra, namely loop theory, in which automated reasoning tools have had perhaps the greatest impact. The paper is self-contained in that we don't assume the reader is familiar with loop theory.

Our goals are twofold. Firstly, we catalogue the loop theory results to date that have been obtained with the assistance of automated theorem provers. Secondly, we lay the groundwork for developing benchmarks for automated theorem provers on genuine research problems from mathematics. Toward that end, we create a library called QPTP (Quasigroup Problems for Theorem Provers) and test the problems on selected automated theorem provers. Note that we don't intend to mirror the TPTP library [SS98]. Rather, we select a representative subset of problems that mathematicians approached by automated reasoning in their research.

We now give a brief outline of the paper.

Section 2 contains a brief introduction to loop theory, with an emphasis on formal definitions (as opposed to motivation, history, applications, etc.). We think this self-contained introduction to loop theory is the right approach for our intended audience: computer scientists interested in applications of automated reasoning in mathematics. For a more rigorous introduction to the theory of loops see [Bel67], [Bru71], or [Pfl90].

[*]This work is a part of the research project MSM 0021620839 financed by MŠMT ČR. The second author was partly supported by the GAČR grant #201/08/P056.

Section 3 contains a catalogue of all the theorems from loop theory that we used in our analysis. Taken together, the papers that contain these theorems—and we give full citations for all of them—constitute a complete list of those results in loop theory that have been achieved to date with the assistance of automated theorem provers.

Section 4 is devoted to the tests of selected theorem provers on these results.

Section 5 contains final thoughts as well as suggested directions for future work.

Additional information on our library, the problem files and the output files may be found on the website

`http://www.karlin.mff.cuni.cz/~stanovsk/qptp`

2 Basic Loop Theory

We call a set with a single binary operation and with a 2-sided identity element 1 a *magma*. There are two natural paths from magmas to groups, as illustrated in Figure 1.

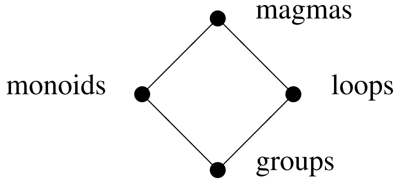

Figure 1: Two paths from magmas to groups.

One path leads through the *monoids*—these are the associative magmas, familiar to every computer scientist. The other path leads through the *loops*—these are magmas in which every equation

$$x \cdot y = z$$

has a unique solution whenever two of the elements x, y, z are specified. Since groups are precisely loops that are also monoids, loops are known colloquially as "nonassociative groups", and via this diagram, they may be thought of as dual to monoids. Many results in loop theory may by regarded as a generalization of results about groups.

As with the class of monoids, the class of loops is too large and general to yield many of its secrets to algebraic inquiry that doesn't focus on narrower subclasses. Here, we simply catalog a few of the most important of these subclasses (the abundant evidence arguing for their importance may be found in many loop theory sources).

First, a comment about notation: we use a multiplication symbol for the binary operation. We usually write xy instead of $x \cdot y$, and reserve $\cdot$ to have lower priority than juxtaposition among factors to be multiplied, for instance, $y(x \cdot yz)$ stands for $y \cdot (x \cdot (y \cdot z))$. We use binary operations $\backslash, /$ of *left* and *right division* to denote the unique solutions of the equation $x \cdot y = z$, ie., $y = x\backslash z$ and $x = z/y$. Loops can thus be axiomatized by the following six identities:

$$x \cdot 1 = x, \quad 1 \cdot x = x,$$

$$x\backslash(xy) = y, \quad x(x\backslash y) = y, \quad (yx)/x = y, \quad (y/x)x = y.$$

Loops without the unit element 1 are refered to as *quasigroups*; in the finite case, they correspond to Latin squares, via their multiplication table.

2.1 Weakening associativity

A *left Bol loop* is a loop satisfying the identity

$$x(y \cdot xz) = (x \cdot yx)z; \tag{lBol}$$

right Bol loops satisfy the mirror identity, namely

$$z(xy \cdot x) = (zx \cdot y)x. \tag{rBol}$$

In the sequel, if we don't specify right or left, and simply write "Bol loop", we mean a left Bol loop.

A left Bol loop that is also a right Bol loop is called *Moufang loop*. Moufang loops are often axiomatized as loops that satisfy any one of the following four equivalent (in loops) identities:

$$x(y \cdot xz) = (xy \cdot x)z, \quad z(x \cdot yx) = (zx \cdot y)x, \quad xy \cdot zx = x(yz \cdot x), \quad xy \cdot zx = (x \cdot yz)x.$$

Generalizing from the features common to both the Bol and the Moufang identities, an identity $\varphi = \psi$ is said to be of *Bol-Moufang type* if: (i) the only operation appearing in $\varphi = \psi$ is multiplication, (ii) the number of distinct variables appearing in φ, ψ is 3, (iii) the number of variables appearing in φ, ψ is 4, (iv) the order in which the variables appear in φ coincides with the order in which they appear in ψ. Such identities can be regarded as "weak associativity". For instance, in addition to the Bol and Moufang identities, examples of identities of Bol-Moufang type include the *extra law*

$$x(y \cdot zx) = (xy \cdot z)x, \tag{extra}$$

and the *C-law*

$$x(y \cdot yz) = (xy \cdot y)z. \tag{C}$$

There are many others, as we shall see. Some varieties of Bol-Moufang type are presented in Figure 2 (for a complete picture, see [PV05]).

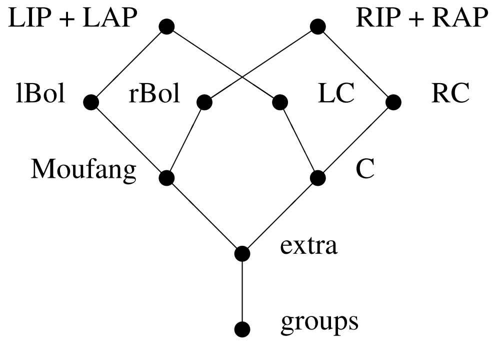

Figure 2: Some varieties of weakly associative loops.

For loops in which each element has a 2-sided inverse, we use x^{-1} to denote this 2-sided inverse of x. In other words,

$$x^{-1}x = xx^{-1} = 1.$$

In Bol loops (hence, also in Moufang loops), all elements have 2-sided inverses. In Moufang loops, inverses are especially well behaved; they satisfy the *anti-automorphic inverse property*

$$(xy)^{-1} = y^{-1}x^{-1}, \tag{AAIP}$$

a familiar law from the theory of groups. Bol loops don't necessarily satisfy the AAIP; in fact, the ones that do (left or right), are Moufang. Dual to the AAIP is the *automorphic inverse property*

$$(xy)^{-1} = x^{-1}y^{-1}. \tag{AIP}$$

Not every Bol loop satisfies the AIP, but those that do are called *Bruck loops*. Bruck loops are thus dual to Moufang loops, with respect to these two inverse properties, in the class of Bol loops.

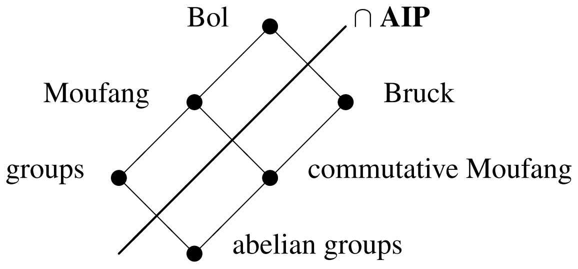

Figure 3: The role of AIP.

A loop is *power associative* if each singleton generates an associative subloop. Bol loops are power associative. Moufang loops satisfy the *flexible* law

$$x \cdot yx = xy \cdot x. \tag{flex}$$

Flexible Bol loops, either left or right, are Moufang. Left Bol loops satisfy both the *left inverse property*

$$x^{-1} \cdot xy = y \tag{LIP}$$

and the *left alternative property*

$$x \cdot xy = xx \cdot y. \tag{LAP}$$

The *right inverse property* (RIP) and the *right alternative property* (RAP) are defined in the obvious ways. The *inverse property* (IP) thus means both the RIP and the LIP, and a loop is called *alternative* if it is both RAP and LAP. Moufang loops and C-loops are alternative and have the inverse property. The *weak inverse property* is given by

$$(yx)\backslash 1 = x\backslash(y\backslash 1). \tag{WIP}$$

2.2 Translations

In a loop Q, the left and right translations by $x \in Q$ are defined by

$$L(x) : y \mapsto xy, \qquad R(y) : x \mapsto xy.$$

The *multiplication group*, $\mathrm{Mlt}(Q)$, of a loop Q is the subgroup of the group of all bijections on Q generated by right and left translations:

$$\mathrm{Mlt}(Q) = \langle R(x), L(x) : x \in Q \rangle.$$

The *inner mapping group* is the subgroup $\mathrm{Mlt}_1(Q)$ fixing the unit element 1. $\mathrm{Mlt}_1(Q)$ is generated by the following three families of mappings, thus rendering the definition equational, and fit for automated theorem provers:

$$T(x) = L(x)^{-1}R(x),$$

$$R(x,y) = R(xy)^{-1}R(y)R(x),$$
$$L(x,y) = L(yx)^{-1}L(y)L(x).$$

If Q is a group, then $\mathrm{Mlt}_1(Q)$ is the group of inner automorphisms of Q. In general, though, $\mathrm{Mlt}_1(Q)$ need not consist of automorphisms. But in those cases in which it does, the loop is called an *A-loop*. Groups and commutative Moufang loops are examples of A-loops.

A subloop invariant to the action of $\mathrm{Mlt}_1(Q)$ (or, equivalently, closed under $T(x)$, $R(x,y)$, $L(x,y)$) is called *normal*. Normal subloops are kernels of homomorphisms, and are thus analogous to normal subgroups in group theory. (In loops, there is no counterpart of the coset definition of a normal subgroup.)

A loop is called *left conjugacy closed* if the conjugate of each left translation by a left translation is again a left translation. This can be expressed equationally as

$$z\cdot yx = ((zy)/z)\cdot zx. \tag{LCC}$$

The definition of *right conjugacy closed* is now obvious, and is given equationally as

$$xy\cdot z = xz\cdot (z\backslash(yz)). \tag{RCC}$$

A *conjugacy closed loop* (CC-loop) is a loop that is both LCC and RCC.

We end this section by defining two classes of loops that are closely related to both Moufang loops and A-loops. *RIF loops* are inverse property loops that satisfy

$$xy\cdot (z\cdot xy) = (x\cdot yz)x\cdot y. \tag{RIF}$$

ARIF loops are flexible loops that satisfy

$$zx\cdot (yx\cdot y) = z(xy\cdot x)\cdot y. \tag{ARIF}$$

2.3 Important subsets and subloops

The *commutant*, $C(Q)$, of a loop Q is the set of those elements which commute with each element in the loop. That is,

$$C(Q) = \{c : \forall x \in Q, cx = xc\}.$$

The commutant of a loop need not be a subloop. Even in those cases when the commutant is a subloop (for instance, in Moufang loops), it need not be normal (of course, the commutant in a group is normal, and in group theory it is called the center, as we shall see).

The *left nucleus* of a loop Q is the subloop given by

$$N_\lambda(Q) = \{a : a\cdot xy = ax\cdot y, \forall x,y \in Q\}.$$

The middle nucleus and the right nucleus, $N_\mu(Q)$ and $N_\rho(Q)$ respectively, are defined analogously; both are subloops. The *nucleus*, then, is the subloop given by

$$N(Q) = N_\lambda(Q)\cap N_\mu(Q)\cap N_\rho(Q).$$

The *center* is the normal subloop given by

$$Z(Q) = N(Q)\cap C(Q),$$

thus coinciding with the language from groups. $C(Q)$ need not have any relationship with $N(Q)$; that is, $C(Q)\cap N(Q) = Z(Q)$ can be trivial. The situation in Bol loops is strikingly different. In a (left) Bol loop

Q, $N_\lambda(Q) = N_\mu(Q)$, and this subloop need not have any relationship with $N_\rho(Q)$, i.e., the intersection can be trivial. Thus, in a Moufang loop, all nuclei coincide, and $N(Q)$ is a normal subloop. Moreover, if Q is Bruck, then $N_\lambda(Q) \leq C(Q)$.

The *commutator*, $[x,y]$ of x and y, in a loop Q is given by

$$xy = yx \cdot [x,y].$$

The *associator*, $[x,y,z]$ of x, y, and z, is given by

$$xy \cdot z = (x \cdot yz) \cdot [x,y,z].$$

The point is that the lack of associativity in loops provides a structural richness, part of which can be captured equationally, thus rendering loops excellent algebraic objects to investigate with automated theorem provers.

3 The Theorems

The present section catalogues all papers in loop theory to date whose results were obtained with the assistance of an automated theorem prover. All proofs were obtained by Prover9 or Otter. The proofs were always translated to human language and usually simplified (none of the papers presents a raw output from a prover), hence none of the results relies on soundness of Otter/Prover9. As far as we know, no automatically generated proof was found to be incorrect during translation.

In some cases, the main results weren't obtained directly with automated theorem provers. Instead, provers were used to prove key technical lemmas, or even just special cases, which in turn helped mathematicians find proofs of the main results. This is explained in more detail below, see, e.g., our description of [AKP06].

We list the papers in chronological order. From each paper, we choose up to five theorems for the QPTP library.

[Kun96a]. This is an important paper, because it was the first to use automated theorem provers in loop theory and, in fact, one of the first noneasy results in mathematics obtained automatically. The theorem says that a quasigroup satisfying any one of the four Moufang laws is, in fact, a loop, i.e., has a unit element. We analyze this result for each of the four Moufang identites. Note that the proof for the third and the fourth Moufang identities can by done relatively easily by hand, while the proof for the first and the second one was only discovered by Otter.

[Kun96b]. This is a sequel to the previous paper. The main result is the determination of which of the Bol-Moufang identities, implies, in a quasigroup, the existence of a unit element. We analyze three of these identities.

[Kun00]. There are many results in this paper proved by automated theorem provers. We analyze the following two: (1) If G is conjugacy closed, with $a,b \in G$ and $ab = 1$, then ba is in the nucleus of G. (2) If G is conjugacy closed, the commutant of G is contained in the nucleus.

[KKP02a]. The main result in this paper is that inverse property A-loops are Moufang; we analyze this result. This was one of the major long-standing open problems in loop theory, and perhaps the most important automated theorem proving success in loop theory. And it marks the point at which the number of loop theorists using automated theorem provers in their work jumped from one to three.

[KKP02b]. There are many results in this paper proved by automated theorem provers; we include the following four: (1) 2-divisible ARIF loops are Moufang, (2) flexible C-loops are ARIF, (3) Moufang loops are RIF, (4) RIF loops are ARIF.

[KK04]. There are many results in this paper in which automated theorem provers helped, e.g., finite nonassociative extra loops have nontrivial centers. We analyze the following result: In an extra loop, z commutes with $[x,y,t]$ if and only if t commutes with $[x,y,z]$ if and only if $[x,y,z][x,y,t]=[x,y,zt]$.

[KKP04]. There are many results in this paper proved by automated theorem provers. We include the following one: in CC-loops, associators are in the center of the nucleus.

[KP04]. The main result in this paper is that commutants of Bol loops of odd order are, in fact, subloops. Obviously, this is not a first order statement, however its proof relies on several lemmas proved by a theorem prover. We analyze the following one: If Q is a Bol loop, and if $a,b \in C(Q)$, then so too are a^2, b^{-1} and a^2b.

[KP05]. The main result in this paper is to give a basis for the variety of rectangular loops which consists of 7 identities, thus improving Krapež's pre-existing basis of 12 axioms [Kra00]. A *rectangular loop* is a direct product of a loop and a rectangular band. A *rectangular band* is a semigroup which is a direct product of a left zero semigroup and right zero semigroup. A *left* (*right*, resp.) *zero semigroup* is a semigroup satisfying $x\cdot y = x$ ($x \cdot y = y$, resp.). We analyze part of this result by showing that the identities

$$x\backslash(xx)=x,\ (xx)/x=x,\ x\cdot(x\backslash y)=x\backslash(xy),\ (x/y)\cdot y=(xy)/y,\ x\backslash(x(y\backslash y))=((x/x)y)/y,$$

$$(x\backslash y)\backslash((x\backslash y)\cdot(zu))=(x\backslash(xz))\cdot u,\ ((xy)\cdot(z/u))/(z/u)=x\cdot((yu)/u)$$

imply each of the following identities (in algebras with three binary operations $\cdot$, $\backslash$, and $/$):

$$(x\backslash y)\backslash((x\backslash y)z)=x\backslash(xz),\ (x/y)\backslash((x/y)z)=x\backslash(xz),\ x(y\backslash(yz))=xz,\ ((xy)/y)z=xz,$$

$$(x\cdot yz)/(yz)=(xz)/z,\ (x(y\backslash z))/(y\backslash z)=(xz)/z,\ (x(y/z))/(y/z)=(xz)/z.$$

[PV05]. The main result of this paper is the systematic classification of all varieties of loops axiomatized by a single identity of Bol-Moufang type, achieved to a large extent automatically. We include a typical result from [PV05]: in loops, the following two identities are equivalent (and thus both axiomatize the so-called variety of LC-loops): $x(y\cdot yz)=(x\cdot yy)z$ and $xx\cdot yz=(x\cdot xy)z$.

[AKP06]. One of the main results in this paper is that in a Bruck loop, elements of order a power of two commute with elements of odd order. Obviously, automated theorem provers can't prove this result directly, as it is a result about infinitely many positive integers. On the other hand, one may use automated theorem provers to generate proofs about *specific* integers, and then use these proofs to help construct the proof of the general result. The three specific cases we analyze here: in a (left) Bruck loop, elements of order 2^2 commute with elements of oder 3, elements of order 2^2 commute with elements of order 3^2, and elements of order 2^4 commute with elements of order 3^2. The three different cases give rise to clear performance differences between the automated theorem provers, as we shall see. We note that this property was used in [AKP06] in a proof of a deep decomposition theorem for Bruck loops. That is, this also was an important success for automated theorem provers in loop theory.

[KK06]. There are many results in this paper proved by automated theorem provers. We analyze the following results: for each c in a power associative conjugacy closed loop, c^3 is WIP (i.e., $c^3(xc)^{-1} = x^{-1}$ for every x), c^6 is extra (i.e., $c^6(x \cdot yc^6) = (c^6x \cdot y)c^6$ for every x, y) and c^{12} is in the nucleus. (Initially, the last property wasn't obtained directly by Prover9. Interestingly, other provers can do it.)

[Phi06]. The main result in this paper is that the variety of power associative, WIP conjugacy closed loops is axiomatized, in loops, by the identities $(xy \cdot x) \cdot xz = x \cdot ((yx \cdot x)z)$ and $zx \cdot (x \cdot yx) = (z(x \cdot xy)) \cdot x$. We analyze this result.

[PV06]. There are many results in this paper proved by automated theorem provers. We analyze the following two: (1) in C-loops, the nucleus is normal, and (2) in a commutative C-loop, if a has order 4 and b has order 9, then $a \cdot bx = ab \cdot x$ (this is one of the cases that led to a proof of the decomposition theorem for commutative torsion C-loops).

[KKP07]. An *F-quasigroup* is a quasigroup that satisfies the following two equations: $x \cdot yz = xy \cdot (x\backslash x)z$ and $zy \cdot x = z(x/x) \cdot yx$. The main result of the paper is that every F-quasigroup is isotopic to a Moufang loop. This was a long-standing open problem—it was the first open problem listed in Belousov's 1967 book [Bel67]. We analyze this result.

[KPV07]. There are many results in this paper proved by automated theorem provers. We analyze the following one: a C-loop of exponent four with central squares is flexible.

[KPV08]. There are many results in this paper proved by automated theorem provers. We include the following one: in a Bol loop, if c is a commutant element, then c^2 is in the left nucleus if and only if c is in the right nucleus.

[PV08]. The purpose of this paper is to find group-like axiomatizations for the varieties of loops of Bol-Moufang type. We include the following typical result: a magma with 2-sided inverses satisfying the C-law is a loop.

[CDKxx]. A *Buchsteiner loop* is a loop that satisfies the following identity: $x\backslash(xy \cdot z) = (y \cdot zx)/x$. These loops are closely related to conjugacy closed loops, and are closely related to loops of so-called Bol-Moufang type [DJxx]. The result from [CDKxx] that we analyze here is that in Buchsteiner loops, fourth powers are nuclear (i.e., $x^4 \in N(Q)$ for every $x \in Q$).

[KKPxx]. The main result in this paper is that in a strongly right alternative ring (with a unit element), the set of units is a Bol loop under ring multiplication, and the set of quasiregular elements is a Bol loop under "circle" multiplication. A *right alternative ring* is a set, R, with two binary operations, $+$ and $\cdot$, such that under $+$, R is an abelian group, under $\cdot$, R is a right alternative magma, and such that $\cdot$ distributes over $+$. A right alternative ring is *strongly right alternative* if $\cdot$ is a right Bol loop. A *unit* in an alternative ring is an element that has a two-sided inverse. The circle operation is given by $x \circ y = x + y + xy$. And finally, an element is *quasiregular* if it has a two-sided inverse under circle, e.g. $x \circ x' = x' \circ x = 0$. We analyze the following technical result: If a has a 2-sided inverse, then $R(a^{-1}) = R(a)^{-1}$ and $L(a)^{-1} = R(a)L(a^{-1})R(a^{-1})$.

[KVxx]. There are many results in this paper proved by automated theorem provers. We analyze the following one: in a commutative RIF loop, all squares are Moufang elements and all cubes are C-elements. An element a is a *Moufang element* if for all x and y, $a(xy \cdot a) = ax \cdot ya$. And it is a *C-element* if for all x and y, $x(a \cdot ay) = ((xa \cdot a)y$.

A remark on TPTP. The intersection of the TPTP and QPTP libraries is empty (by now). The only loop theory problem in TPTP is the equivalence of the four Moufang identities (GRP200 – GRP206). This result is included in the book [MP96], which demonstrates the power of Otter in selected areas of mathematics (the other loop theory problems in the book are several single axioms).

4 Benchmark tests

In the present section, we analyze the problems in the QPTP library by running them on selected automated theorem provers. Based on the results of the CASC competition in recent years [SS06], we chose the following five provers: E [Sch02], Prover9 [McC05], Spass [S], Vampire [RV02] and Waldmeister [Hil03].

We ran each prover on each file twice: with 3600 and 86400 seconds time limit (1 hour and 1 day). The problems from the library were translated to the TPTP syntax, and the input files for the provers were generated by the tptp2X tool. We ran the provers with their default settings, and we didn't tune any of the input files for a particular prover, thus obtaining conditions similar to the CASC competition.

Our results are presented in Figure 4. The names of the problems start with the code of the paper in the bibliography followed by the code of the selected result. In the case when there is no single obvious way to formalize the statement in first order theory, alternative axiomatizations are given. (For details, see the QPTP website.) Running time (i.e., the time it took to find a proof) is displayed in rounded seconds; a blank space means timeout, cross means that the problem is not equational and thus ineligible for Waldmeister. Running times over 360s (the time limit of the last CASC) are displayed in bold, running times over 1 hour in italic.

The total number of problems in QPTP is 80, of which 68 are equational. 71 problems were solved by at least one prover, 38 by all of them. The overall performance of the provers is summarized in Figure 5.

In our study, Waldmeister performed better than the other four provers on equational problems. The performances of E, Prover9 and Vampire seem to be similar (incomparable in the strict sense), although E can be quite fast on some difficult problems. Spass seems to be well behind the other provers.

In order to minimize bias in our study, we ignored basic parameter settings. While some provers work fully automatically (e.g., Vampire), other actually have *no* default setting (e.g., Waldmeister). In our case, "default" is defined by the output of tptp2X. In particular, this means the set(auto) mode for Prover9 and some explicit term ordering for Waldmeister.

In fact, term ordering is perhaps the most influential parameter. Waldmeister's default ordering is KBO with weights 1, while Prover9's default is LPO. If Prover9 is manually reset to KBO, it proves six additional files, but fails for two files that were proved with LPO; this way, Prover9's performance becomes closer to Waldmeister's. Another influential parameter is symbol ordering. Prover9 chooses a relatively smart one (inverse $>$ left/right division $>$ multiplication), while Waldmeister gets from tptp2X an alphabetical one (inverse $>$ left division $>$ multiplication $>$ right division). When reset to the smarter choice, it's on average much faster, although it fails to prove any additional problems. (Note that perhaps the smartest choice is division $>$ inverse $>$ multiplication.) Indeed, parameter setting deserves much greater attention, but this is beyond the scope of the present study.

file/prover	E 0.999-006	Prover9 1207	Spass 3.0	Vampire 8.0	Waldmeister 806
AKP06_1	0	11	**459**	6	0
AKP06_2	16	**1110**			74
AKP06_3					
CDKxx_1a					
CDKxx_1b					
CDKxx_1c					
KK04_1					*29919*
KK04_2					*29387*
KK04_3					*10322*
KK06_1a	57			**507**	59
KK06_1b	**922**				**592**
KK06_1c	*46277*				**570**
KK06_1d	*53534*				**560**
KK06_1e	*46687*				**554**
KKP02a_1	**3023**	*26735*			x
KKP02a_1alt1	**848**	*36852*		**553**	205
KKP02a_1alt2	**848**	*35016*		**500**	208
KKP02a_1alt3	**1001**	*24832*		**550**	213
KKP02a_1alt4	**1018**	*24242*		**584**	202
KKP02b_1	4	120	99	97	x
KKP02b_1alt1	9	205		**488**	8
KKP02b_1alt2	3	147	**413**	**484**	8
KKP02b_1alt3	9	208	*56918*	**491**	9
KKP02b_1alt4	9	190	*53195*	**485**	9
KKP02b_2	0	0	**475**	10	1
KKP02b_3	0	0	0	0	2
KKP02b_4a	31	**1462**		138	4
KKP02b_4b	0	0	0	0	0
KKP04_1a					
KKP04_1b					
KKP04_1c					
KKP04_2				**2856**	**580**
KKP07_1					**2265**
KKPxx_1	2	0	3	8	0
KKPxx_2a					
KKPxx_2b					
KP04_1	0	0	0	0	0
KP04_2	0	0	0	0	0
KP04_3	7	73	*72463*	270	2
KP05_1a	0	0	0	0	0
KP05_1b	0	0	0	0	0
KP05_1c	0	0	0	0	0
KP05_1d	0	0	0	0	0
KPV07_1	0	0	0	0	0
KPV08_1	0	0	0	0	0
KPV08_2	0	0	0	0	0
Kun00_1a	352	*7088*		*12482*	**705**
Kun00_1b	353	*7536*		*15412*	**736**
Kun00_1c	353	**3264**		*19762*	**706**
Kun00_1alt1		*38587*			**690**
Kun00_2	0	0	0	0	0
Kun96a_1	56	75		258	x
Kun96a_1alt1	128	112		218	3
Kun96a_1alt2	9	68		238	3
Kun96a_2	57	**1256**		285	x
Kun96a_2alt1	8	**398**		284	3
Kun96a_2alt2	51	**1282**		164	3
Kun96a_3	0	0	0	0	x
Kun96a_4	0	0	0	0	x
Kun96b_1	0	0	1	0	x
Kun96b_2	0	1	9	0	x
Kun96b_3	0	19	161	5	x
Kun96b_3alt1	0	7	125	28	0
Kun96b_3alt2	0	5	148	43	0
KVxx_1		357		**1692**	49
KVxx_2		**1705**		**3172**	95
Phi06_1a	72	29		14	21
Phi06_1b	46	2	*8632*	6	17
Phi06_2a	0	118	41	1	0
Phi06_2b	0	1	0	**473**	0
Phi06_2c	0	0	0	0	x
Phi06_3	0	41	**857**	9	0
PV05_1	0	1	19	6	0
PV05_2		9	5	1	0
PV06_1a	0	0	0	0	0
PV06_1b	0	0	0	0	0
PV06_1c	0	0	0	0	0
PV06_2	34	17		4	0
PV08_1a	0	0	1	0	x
PV08_1b	0	0	0	10	x

Figure 4: Detailed results.

prover	E 0.999	Prover9 1207	Spass 3.0	Vampire 8.0	Waldmeister 806
proofs in 360s	53	46	31	44	46
proofs in 3600s	59	53	35	57	56
proofs in 86400s	62	61	39	60	59
timeouts	18	19	41	20	9

Figure 5: Summary.

Finally, the reader may wonder about those theorems on which all provers were unsuccessful (these are indicated by blank entries in Figure 4). After all, these are theorems that were first proved with the assistance of an automated theorem prover (which was the sole criterion for inclusion in our study). Why were none of the provers able to find proofs in our study? Firstly, we didn't tune the input files. And, perhaps more importantly, we didn't use any advanced techniques in our study (e.g., Prover9's powerful hints strategy), which is often the way to obtain a new mathematical result.

5 Conclusions

While we hope our results are interesting to automated reasoning researchers (especially since they involve problems from an active area of mathematical research), they may not be *surprising* to these same researchers, informed as these researchers are by the CASC results over the past ten years. Our results, though, might surprise loop theorists, who are less familiar with most of the provers in our study. But again, we stress that some of these loop theory results were originally obtained using advanced Otter/Prover9 techniques such as the hints strategy, or sketches [Ver01]. Could these be implemented in other provers?

Since the various automated theorem provers have different strengths and weaknesses, loop theorists could profit by using a suite of theorem provers in their investigations. For instance, the result in [KKP07] was originally derived as a series of results, a number of steps eventually leading to the main theorem. In our study, Waldmeister proved it from scratch in 40 minutes. To state the obvious: some theorems will be missed if one uses only one automated theorem prover. On the other hand, the actual proofs themselves are, of course, of great importance, and the various automated theorem provers differ greatly in this regard. Some provers don't even give the proof, they simply indicate that they've found one, while others, notably Prover9, produce relatively readable proofs, and even include tools to simplify them further.

There are clearly many opportunities for future work. We intend to keep our catalogue of loop theory results as up-to-date as possible. We eventually hope to test our files on more automated theorem provers; in particular, on some instantiation based ones, to check the hypothesis that those are weak on algebraic problems (that always require a lot of computation with equality). Another obvious direction for future work is to analyze results in other domains, for instance quasigroups and other nonassociative algebras.

It would be immensely useful to compare the various provers' performance by using input files and techniques designed to obtain the best performance for each particular prover. In particular, and for example, what are the best first order descriptions of properties like "existence of a unit" (the formula $\exists x \forall y (xy = y \ \& \ yx = x)$, or the identities $(x/x)y = y, y(x/x) = y$, or the identities $(x\backslash x)y = y, y(x\backslash x) = y$), or of the Moufang property (in what ways might the fact that the four defining identities of this variety are equivalent impact "best performance" strategies amongst the various provers?).

A possible new direction of exploiting automated reasoning to prove new theorems about loops could be, first, to create a knowledge base of definitions and theorems in loop theory (those that can be expressed within the first order theory of loops) and then to apply tools for reasoning in large theories. Particularly, such a knowledge base would substantially differ from other recent projects such as the MPTP [Urb04]. The QPTP library can be viewed as the zeroth step towards such a library.

Acknowledgement. We thank Michael Kinyon and Bob Veroff for carefully reading, and then commenting on, an earlier version of this paper.

[AKP06] M. Aschbacher, M.K. Kinyon, and J.D. Phillips, Finite Bruck loops, *Transactions of the American Mathematical Society*, **358** (2006), 3061–3075.

[Bel67] V.D. Belousov, Foundations of the Theory of Quasigroups and Loops, Nauka, Moscow, 1967 (in Russian).

[Bru71] R. H. Bruck, A Survey of Binary Systems, third printing, corrected, *Ergebnisse der Mathematik und ihrer Grenzgebiete*, New Series **20**, Springer-Verlag, 1971.

[CDKxx] P. Csorgö, A. Drápal, M.K. Kinyon, Buchsteiner loops, submitted.

[DJxx] A. Drápal, P. Jedlička, On loop identities that can be obtained by nuclear identification, submitted.

[Hil03] T. Hillenbrand, Citius altius fortius: Lessons Learned from the Theorem Prover Waldmeister, in Dahn I., Vigneron L., Proceedings of the 4th International Workshop on First-Order Theorem Proving (Valencia, Spain), Electronic Notes in Theoretical Computer Science 86.1, Elsevier Science, 2003. `http://www.waldmeister.org`

[Kra00] A. Krapež, Rectangular loops, *Publ. Inst. Math. (Beograd) (N.S.)*, **68(82)** (2000), 59–66.

[Kun96a] K. Kunen, Moufang quasigroups *Journal of Algebra*, **183** (1996) no. 1, 231–234.

[Kun96b] K. Kunen, Quasigroups, loops, and associative laws *Journal of Algebra*, **185** (1996) no. 1, 194–204.

[Kun00] K. Kunen, The structure of conjugacy closed loops, *Transactions of the American Mathematical Society*, **352** (2000) no. 6, 2889–2911.

[KK04] M.K. Kinyon and K. Kunen, The structure of extra loops *Quasigroups Related Systems*, **12** (2004), 39–60.

[KK06] M.K. Kinyon and K. Kunen, Power-associative, conjugacy closed loops, *Journal of Algebra*, **304** (2006), no. 2, 679–711.

[KKP02a] M.K. Kinyon, K. Kunen, and J.D. Phillips, Every diassociative A-loop is Moufang, *Proceedings pf the American Mathematical Society*, **17** 130 (2002), 619–624.

[KKP02b] M.K. Kinyon, K. Kunen, and J.D. Phillips, A generalization of Moufang and Steiner loops, *Algebra Universalis*, **48** (2002), 81–101.

[KKP04] M.K. Kinyon, K. Kunen, and J.D. Phillips, Diassociativity in conjugacy closed loops, *Communications in Algebra*, **32** (2004), 767–786.

[KKP07] T. Kepka, M.K. Kinyon, and J.D. Phillips, The structure of F-quasigroups, *Journal of Algebra*, 317 (2007), 435–461.

[KKPxx] M.K. Kinyon, K. Kunen, and J.D. Phillips, Strongly right alternative rings and Bol loops, *Publicationes Mathematicae Debrecen*, submittted.

[KP04] M.K. Kinyon and J.D. Phillips, Commutants of Bol loops of odd order, *Proceedings of the American Mathematical Society*, **132** (2004), 617–619.

[KP05] M.K. Kinyon and J.D. Phillips, Rectangular quasigroups and rectangular loops, *Computers and Mathematics with Applications* **49** (2005), **11–12**, 1679–1685.

[KPV07] M.K. Kinyon, J.D. Phillips, and P. Vojtěchovský, C-loops: extensions and constructions, *Journal of Algebra and its Applications*, **6 (1)**, (2007), 1–20.

[KPV08] M.K. Kinyon, J.D. Phillips, and P. Vojtěchovský, When is the commutant of a Bol loop a subloop? *Transactions of the American Mathematical Society*, 360 (2008), no. 5, 2393–2408.

[KVxx] M.K. Kinyon and P. Vojtěchovský, Primary decompositions in varieties of commutative diassociative loops, submitted.

MP96 W.W. McCune and R. Padmanabhan, *Automated Deduction in Equational Logic and Cubic Curves*, Springer-Verlag, 1996.

[McC03] W. W. McCune, *OTTER 3.3 Reference Manual and Guide*, Argonne National Laboratory Technical Memorandum ANL/MCS-TM-263, 2003;
`http://www.mcs.anl.gov/AR/otter/`

[McC05] W. W. McCune, *Prover9*, automated reasoning software, and *Mace4*, finite model builder, Argonne National Laboratory, 2005.
`http://www.prover9.org`

[NV] G. P. Nagy and P. Vojtěchovský, *LOOPS: a package for GAP* 4,
available at `http://www.math.du.edu/loops`

[Pfl90] H. O. Pflugfelder, Quasigroups and Loops: Introduction, *Sigma Series in Pure Mathematics* **7**, Heldermann Verlag Berlin, 1990.
[Phi03] J.D. Phillips See Otter digging for algebraic pearls, *Quasigroups and Related Systems*, **10** (2003), 95–114.
[Phi06] J.D. Phillips, A short basis for the variety of WIP PACC-loops, *Quasigroups and Related Systems*, **14** (2006), 73–80.
[PV05] J.D. Phillips and P. Vojtěchovský, The varieties of loops of Bol-Moufang type, *Algebra Universalis*, **54 (3)** (2005), 259–271.
[PV06] J.D. Phillips and P. Vojtěchovský, C-loops: an introduction, *Publicationes Mathematicae Debrecen*, **68/1–2** (2006), p. 115–137.
[PV08] J.D. Phillips and P. Vojtěchovský, A scoop from groups: new equational foundations for loops, *Commentationes Mathematicae Universitatis Carolinae*, 49/2 (2008), 279–290.
[RV02] A. Riazanov, A. Voronkov, The Design and Implementation of Vampire, *AI Communications* 15(2-3) (2002), pp.91-110.
[S] `http://spass.mpi-sb.mpg.de/`
[Sch02] S. Schulz, E: A Brainiac Theorem Prover, *AI Communications* 15(2-3), 111–126.
`http://www4.informatik.tu-muenchen.de/~schulz/WORK/eprover.html`
[SS98] G. Sutcliffe, C. Suttner, The TPTP Problem Library: CNF Release v1.2.1, *Journal of Automated Reasoning*, 21/2 (1998), 177–203.
[SS06] G. Sutcliffe, C. Suttner, The State of CASC, *AI Communications* 19/1 (2006), 35-48.
[Urb 04] Josef Urban: MPTP - Motivation, Implementation, First Experiments. *J. Autom. Reasoning* 33(3-4): 319-339 (2004)
[Ver01] R. Veroff, Solving open questions and other challenge problems using proof sketches, *J. Automated Reasoning* 27(2) (2001), 157–174.
[ZZ] J. Zhang, H. Zhang, SEM: a System for Enumerating Models,
`http://www.cs.uiowa.edu/~hzhang/sem.html`