Basic Logic, SMT solvers and finitely generated varieties of GBL-algebras

Peter Jipsen

Chapman University, Orange, California

TACL 2013, July 29, Vanderbilt University

Some non-classical logics

Intuistionistic Logic - non-classical logic of constructive provability

Łukasiewicz Logic - non-classical many-valued logic

Modal Logic - classical logic extended with modalities

Generalized Basic Logic is a common generalization of Intuistionistic Logic and Łukasiewicz Logic

Outline

1. Propositional Basic Logic formulas can be decided efficiently with SMT-solvers

2. The lattice of finitely generated varieties of (G)BL-algebras can be described

3. The n-element frames of generalized basic logic are in one-to-one correspondence with the n-element Kripke frames of the modal logic S4

Basic Logic algebras

Consider any **continuous commutative order-preserving monoid** operation on $[0, 1]$ with unit 1 (also called **t-norm**)

E.g. $xy = \min(x, y)$

or $xy = \max(x + y - 1, 0)$

or $xy = x \cdot y$ (multiplication)

Define $x \to y = \sup\{z \in [0, 1] : xz \leq y\}$

The algebra $A = ([0, 1], \min, \max, \cdot, 1, \to, 0)$ is a BL-algebra

BL is the variety generated by all these algebras

Axiomatizing BL-algebras

Hajek [1998] gave a finite equational axiomatization that was shown to be complete by Cignoli, Esteva, Godo, Torrens [2000]

BL is the variety of commutative residuated lattices with bottom 0 such that

$x \wedge y = x(x \to y)$ and $(x \to y) \vee (y \to x) = 1$

Contains all **Boolean Algebras** (add $xx = x = (x \to 0) \to 0$)

Gödel Algebras: variety gen by [0,1] with $xy = \min(x, y)$

MV-algebras: var gen by [0,1] with $xy = \max(x + y - 1, 0)$

Product Algebras: variety gen by [0,1] with $xy = x \cdot y$

All these varieties have **decidable equational theories**

But how do we decide a particular equation **in practice**?

SAT-solvers

SAT stands for *satisfiability* of Boolean formulas

Given a Boolean formula φ with propositional variables $p_1, \ldots, p_n$

decide if there is an assignment $h : \{p_1, \ldots, p_n\} \to \{T, F\}$ such that

h extended homomorphically to all formulas makes $h(\varphi) = T$

SAT was the first problem proved to be NP-complete

i.e., there is a nondeterministic Turing machine that decides SAT in polynomial time and every other problem that can be decided in nondeterministic polynomial time has a polynomial time reduction to a SAT problem

SMT-solvers

SMT stands for *satisfiability modulo theories*

Combines SAT-solving with other decision procedures for fragments of first-order logic and arithmetic

SMT-solvers were developed in computer science for static analysis of programs

Input is a (limited) choice of a decidable theory and a list of Boolean combinations of atomic formulas in the signature of this theory

Quantifier-free decidable theories

QF_LRA **quantifier free linear real number arithmetic** with $+, -, <, =$

e.g. $\textbf{not}(0 > x + y \textbf{ or } x + y > 5) \textbf{ and } (x + x - y - y = 1)$

QF_RA is like QF_LRA but also allows multiplication, division

SMT-solvers decide if there exists an assignment of real numbers to the variables in the list of formulas such that all the formulas are true in $\mathbb{R}$; return assignment if it exists

How SMT-solvers work

Basic idea: replace atomic formulas by Boolean variables, call a SAT-solver

if the Boolean formulas are **not satisfiable**, return **F**

else use each possible Boolean assignment to generate a list of linear atomic formulas and call a **Linear Programming package**

if an assignment is found, return it, but if none of the Boolean assignments work, return **F**

SMT-solver input for abelian ℓ-groups

Easy, the variety of abelian ℓ-groups is generated by $(\mathbb{R}, \min, \max, +, -, 0)$

SMT_LIB2 is a standard LISP-like language for SMT-solver input

```
;Testing abelian l-group equations in SMT
(set-logic QF_LRA)
(define-fun wedge ((x Real) (y Real)) Real (ite (> x y) y x))
(define-fun vee ((x Real) (y Real)) Real (ite (> x y) x y))
(declare-const x Real)
(declare-const y Real)
(assert (> (vee (+ x x) (+ y y)) (+ (vee x y) (vee x y))))
```
; test if $(x + x) \vee (y + y) \leq (x \vee y) + (x \vee y)$ is an identity
```
(check-sat)
```

SMT-solver input for infinitely-valued logics

The idea of using SMT-solvers for logics based on intervals of the real numbers is from the following paper:

C. Ansótegui, M. Bofill, F. Manyà and M. Villaret, ***Building automated theorem provers for infinitely-valued logics with satisfiability modulo theory solvers***, in Proceedings, IEEE 42nd International Symposium on Multiple-Valued Logic. ISMVL 2012, 25–30.

They give examples of SMT-LIB2 code for **Lukasiewicz logic** and **product logic**

SMT-solver input for MV-algbras

The variety of MV-algebras is $HSP(([0,1], \wedge, \vee, \cdot, 1, 0, \to))$

```
;Testing MV-algebra equations in SMT
(set-logic QF_LRA)
(define-fun wedge ((x Real) (y Real)) Real (ite (> x y) y x))
(define-fun vee ((x Real) (y Real)) Real (ite (> x y) x y))
(define-fun oplus ((x Real) (y Real)) Real (wedge (+ x y) 1))
(define-fun cdot ((x Real) (y Real)) Real (vee (- (+ x y) 1) 0))
(define-fun neg ((x Real)) Real (- 1 x))
(define-fun to ((x Real) (y Real)) Real (wedge 1 (- (+ 1 y) x)))
(declare-const x Real) (assert (<= 0 x)) (assert (<= x 1))
(declare-const y Real) (assert (<= 0 y)) (assert (<= y 1))
(assert (< (to (vee (cdot x x) (cdot y y)) (cdot (vee x y) (vee
x y))) 1))
```

; test if $(x^2 \vee y^2) \to (x \vee y)^2 < 1$ is satisfiable

```
(check-sat)
```

Other standard Basic Logic algebras

For Gödel algebras redefine fusion as min(x,y).

```
(define-fun cdot ((x Real) (y Real)) Real (ite (> x y) y x))
```

For product algebras use

```
(define-fun cdot ((x Real) (y Real)) Real (ite (> x y) y x))
(declare-const x Real) (assert (<= x 0));
(declare-const x Real) (assert (<= x 0));
```

and do a translation to the formula that adds an extra variable z (for bottom)

replacing variable x by $x \vee z$ and subterms $s \cdot t$ by $s \cdot t \vee z$

Prop 7.4 in Galatos, Tsinakis (2005) Generalized MV-algebras

Checking identities in BL-algebras

To decide propositional basic logic with an SMT-solver requires the following result of Agliano Montagna 2003 (see also Aguzzoli and Bova 2010).

Theorem

Let $A_n = \bigoplus_{i=0}^{n}[0,1]$ be the ordinal sum of $n+1$ unit-interval MV-algebras, and let $\mathcal{V}_n$ be the variety generated by all n-generated BL-algebras. Then $\mathcal{V}_n = HSP(A_n)$, hence an n-variable BL-identity holds in A_n if and only if it holds in all BL-algebras.

By constructing the algebra A_n of the above result within the SMT language, one obtains an effective means of checking n-variable BL-identities.

Checking identities in BL-algebras

The universe for A_n is taken to be the interval $[0, n+1]$
The definition of fusion and implication are

$$x \cdot y = \begin{cases} \max(x + y - 1 - \lfloor y \rfloor, \lfloor x \rfloor) & \text{if } \lfloor x \rfloor = \lfloor y \rfloor \\ \min(x, y) & \text{otherwise} \end{cases}$$

$$x \to y = \begin{cases} n + 1 & \text{if } x \leq y \\ y & \text{if } \lfloor y \rfloor < \lfloor x \rfloor \\ \min(1 + y - x + \lfloor x \rfloor, 1 + \lfloor y \rfloor) & \text{otherwise} \end{cases}$$

A straightforward SMT-LIB2 implementation of these operations uses $n+1$ cases, so the formula does become long even for small values of n

Below we give the implementations for $n = 1$ and $n = 2$, which can be used to check 1-variable and 2-variable BL-identities

Checking identities in BL-algebras

$n = 1$:

(define-fun cdot ((x Real) (y Real)) Real (ite (and (< x 1) (< y 1)) (vee (- (+ x y) 1) 0) (ite (and (>= x 1) (>= y 1)) (vee (- (+ x y) 2) 1) (wedge x y))))

(define-fun to ((x Real) (y Real)) Real (ite (<= x y) 2 (ite (and (>= x 1) (< y 1)) y (wedge 1 (- (+ 1 y) x)))))

$n = 2$:

(define-fun cdot ((x Real) (y Real)) Real (ite (and (< x 1) (< y 1)) (vee (- (+ x y) 1) 0) (ite (and (>= x 1) (< x 2) (>= y 1) (< y 2)) (vee (- (+ x y) 2) 1) (ite (and (>= x 2) (>= y 2)) (vee (- (+ x y) 3) 2) (wedge x y)))))

(define-fun to ((x Real) (y Real)) Real (ite (<= x y) 3 (ite (and (< x 1) (< y 1)) (+ (- 1 x) y) (ite (and (<= 1 x) (< x 2) (<= 1 y) (< y 2)) (+ (- 2 x) y) (ite (and (<= 2 x) (<= 2 y)) (+ (- 3 x) y) y)))))

Automating the translation

A Python program is used to parse a LaTeX BL-algebra identity

A SMT-LIB2 file is generated using $\cdot$ and $\rightarrow$ of A_n

The python program then calls an SMT-solver with the file as input

The result is analyzed and the truth value is returned

If the identity fails, an assignment in $[0, n]$ can be obtained

Demo

A **Generalized Basic Logic algebra** (GBL-algebra) is a residuated lattice $(A, \wedge, \vee, \cdot, 1, \backslash, /)$ that satisfies the quasi-identities

$$x \leq y \quad \Longrightarrow \quad x = (x/y)y \text{ and } x = y(y\backslash x)$$

or equivalently the identities

$$x \wedge y = ((x \wedge y)/y)y = y(y\backslash(x \wedge y)).$$

(**Residuated** means: $xy \leq z \iff y \leq x\backslash z \iff x \leq z/y$)

Integral GBL-algebras are defined by requiring $x \leq 1$

Add **bottom** 0, **commutativity**, and $(x \to y) \vee (y \to x) = 1$

get back to Hajek's **Basic Logic algebras**

Open Problem: Is the equational theory of GBL-algebras decidable?

Finite GBL-algebras

They are bounded, hence integral GBL-algebras

[J. & Montagna 06] Finite GBL-algebras are commutative

[J. & Montagna 09] Finite GBL-algebras are **poset products**

of **Wajsberg chains** $W_n = (\{0, a^{n-1}, \ldots, a^3, a^2, a, 1\}, \cdot, 1, \to)$

A poset product is a subalgebra of a direct product over a partially ordered index set

Building all finite GBL-algebras

Let D be a finite distributive lattice with $J(D)$ the set of join-irreducibles

A partition $C_1, \ldots, C_n$ of $J(D)$ consists of **isolated chains** if each C_i is a chain and
$\forall i \neq j[\exists x \in C_i, y \in C_j\,(x < y) \implies \forall x \in C_i, y \in C_j\,(x < y)]$

Theorem: If the coarsest partition of $J(D)$ into isolated chains has n blocks then the number of GBL-algebras with reduct D is $2^{|J(D)|-n}$.

Proof idea: The top element of an isolated chain is always idempotent

The remaining $|J(D)| - n$ elements are possible choices for idempotents

Each idempotent element together with the (possibly empty) chain of nonidempotent elements immediately below it and a

Each algebra is **subdirectly irreducible** iff $J(D)$ has a **top**

GBL-algebras are **congruence distributive**

Hence we can construct lattice of **finitely generated subvarieties**

Here we only consider the case of BL-algebras, so D is a chain

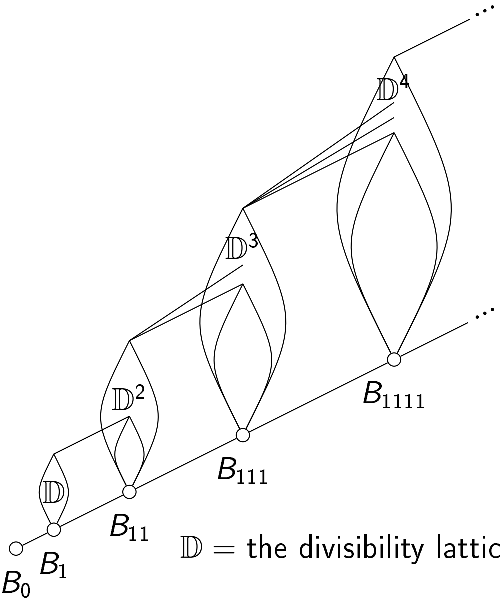
W_m is a **subalgebra** of W_n iff $m|n$

Therefore the varieties $Var(W_n)$, ordered by inclusion, form the **divisibility lattice** $\mathbb{D}$

The lattice of all finitely generated subvarieties of MV-algebras is isomorphic to the downset lattice of $\mathbb{D}$ [Komori 1981]

Theorem. The poset of **finitely generated join irreducible BL-varieties** is isomorphic to $\mathbb{D}^* = \bigcup_{n=0}^{\infty} \mathbb{D}^n$ with the order on $\mathbb{D}^*$ extending the pointwise divisibility order on each component as follows: The order relation $(a_1, \dots, a_m) \leq (b_1, \dots, b_n)$ is a **covering relation** if and only if either

- $m = n$ and $(b_1, \dots, b_n) = (a_1, \dots, a_{i-1}, pa_i, a_{i+1}, \dots, a_n)$ for some prime p and a unique $i \leq n$, or
- $m + 1 = n$ and $(b_1, \dots, b_n) = (a_1, \dots, a_{i-1}, 1, a_i, \dots, a_m)$ for some $i \in \{2, \dots, n\}$



$\mathbb{D}$ = the divisibility lattice on $\mathbb{Z}^+$

Duals of finite GBL-algebras

Let A be a finite GBL-algebra and partition $J(A)$ into isolated chains such that only the top of each chain is an idempotent

These chains are completely **determined** by their cardinality

so the dual of a finite GBL-algebra is a **finite preorder** $\sqsubseteq$

the blocks of the equivalence relation $\sqsubseteq \cap \sqsupseteq$ contain the nonzero elements of each Wajsberg chain

Homomorphisms between finite GBL-algebras correspond to certain p-morphisms between the preorders

hence the HS-poset of finite subdirectly irreducible GBL-algebras can be obtained from these preorder duals

Connections to S4 modal logic

A preorder also determines a Kripke model of the modal logic S4, so every finite GBL-algebra can be mapped to a finite closure algebra

the homomorphisms between finite GBL-algebras are homomorphisms between the corresponding closure algebras

the converse does not hold

Theorem

There is a functor, faithful on objects, from the category of finite GBL-algebras to the category of finite closure algebras and hence also to its dual category of preorders with p-morphisms.

Can extend this functor to larger categories, such as the category of complete perfect n-potent GBL-algebras

Some References

P. Agliano and F. Montagna, *Varieties of BL-algebras I: general properties*, Journal of Pure and Applied Algebra, 181 (2003), 105–129

S. Aguzzoli and S. Bova, *The free n-generated BL-algebra*, Annals of Pure and Applied Logic 161 (2010), 1144–1170

C. Ansótegui, M. Bofill, F. Manyà and M. Villaret, *Building automated theorem provers for infinitely-valued logics with satisfiability modulo theory solvers*, in Proceedings, IEEE 42nd International Symposium on Multiple-Valued Logic. ISMVL 2012, 25–30

N. Galatos and C. Tsinakis, *Generalized MV-algebras*, Journal of Algebra, 283(1) (2005), 254–291

P. Jipsen and F. Montagna, *The Blok-Ferreirim theorem for normal GBL-algebras and its application*, Algebra Universalis, 60 (2009), 381–404

Thank You