Model Checking with Multi-Valued Logics

Glenn Bruns Patrice Godefroid

Bell Laboratories, Lucent Technologies, {grb,god}@bell-labs.com

Abstract. In multi-valued model checking, a temporal logic formula is interpreted relative to a structure not as a truth value but as an element of a lattice. Applications of multi-valued model checking include abstraction techniques, temporal logic query checking, and the analysis of conflict in multi-view specifications. In this paper we simplify and extend existing work on algorithms for multi-valued model checking. We show how to reduce multi-valued model checking with any distributive DeMorgan lattice to standard, two-valued model checking. From this reduction we derive complexity bounds for multi-valued model checking for various temporal logics. We also present a direct, automata-theoretic algorithm for multi-valued model checking with logics as expressive as the modal mu-calculus. As part of showing the correctness of this algorithm, we present new, basic results about extended alternating automata, a generalization of standard alternating automata.

1 Introduction

Model checking is an effective method for checking the correctness of concurrent reactive systems. Recently, several new applications, such as abstract model checking [2], temporal logic query checking [4] and the analysis of multi-view specifications [8], have prompted the study of a new type of model checking, called *multi-valued model checking*. In multi-valued model checking, one interprets a temporal logic formula on a *multi-valued Kripke structure*, which is like a Kripke structure except that an atomic proposition is interpreted at a state as a lattice element, not a truth value. The meaning of a temporal logic formula at a state in a multi-valued Kripke structure is then also given as a lattice element.

Each application of multi-valued model checking has a corresponding lattice or class of lattices. Figure 1 shows some example lattices. Lattice L_2 is a linear ordering of the standard truth values. Model checking with this lattice gives ordinary, two-valued model checking. In lattice L_3, element 1 represents truth, element 0 represents falsity, and element $1/2$ represents "unknown whether true or false" [15, 22]. The interpretation of a formula as value $1/2$ means roughly that the model did not contain enough information to allow a definitely true or definitely false result. Model checking with this lattice can be used to analyze incomplete models of large or complex systems [2]. Lattice L_{fuz} is like L_3 but more degrees of certainty are captured. Fuzzy logic [19] can be regarded as a logic over a lattice of this form containing the reals from 0 to 1. In lattice $L_{2,2}$, an element represents the collective interpretation of two separate parties. A

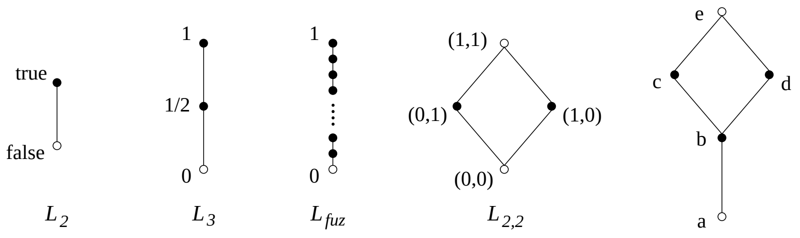

Fig. 1. Some Distributive Lattices

model check with result $(1, 0)$ means that the first party interprets the formula as true while the second party interprets the formula as false. Model checking with lattices like $L_{2,2}$ can be used to analyze whether conflict will arise when multiple specifications or views are combined [8, 14]. Lattices can also be defined in which each element is a set of propositional formulas. Model checking with such lattices has been used for temporal logic query checking [5, 4, 9].

There are two main kinds of algorithms for multi-valued model checking. The first uses reduction: a multi-valued model checking problem is reduced to a set of standard, two-valued model checking problems [3, 16, 14]. For example, in the case of lattice L_3, a model checking problem for a Kripke structure over L_3 can be reduced to two model checking problems for Kripke structures over L_2. The second is direct: multi-valued model checking is performed directly using specialized algorithms. Various model checking algorithms, such as those based on automata or BDDs, can be adapted to work with multi-valued Kripke structures.

Both approaches have advantages. An advantage of reduction is that existing model checkers can be easily adapted for the multi-valued case. Also, as improvements are made to existing model checkers, the benefits can immediately be realized for the multi-valued case. The advantage of the direct approach is that it works in a more "on-demand" manner than the reduction approach. This point is discussed further in Section 6.

This paper describes new algorithms for multi-valued model checking, both by reduction and directly. For checking by reduction, we show that, for a finite distributive lattice, the number of standard model checks required is equal to the number of join-irreducible elements of the lattice in the worst case. From a multi-valued Kripke structure over a finite distributive lattice, we show how a standard Kripke structure can be derived for each join-irreducible element of the lattice, and how the results of model checking on each of these Kripke structures can be combined to give a result for the multi-valued model check. From this general reduction procedure, we also obtain complexity bounds for the multi-valued model-checking problem for various temporal logics.

For direct checking, we use extended alternating automata (EAA) [4], a generalization of alternating automata (AA) used in standard model checking. In model checking applications of AA (e.g., [18]), an input tree of the automaton has nodes that are labelled with sets of atomic propositions, and a run of the automaton has no value associated with it. With EAA, the nodes of the input tree are labelled with functions mapping atomic propositions to elements of a lattice, and a run has an associated value. We recall the definition of EAA, study their properties, and show how to use EAA for multi-valued model checking.

Our results streamline and significantly extend existing work on multi-valued model checking. Concerning reduction, our technique works for all finite, distributive lattices, while existing work ([3],[16]) covers only selected sub-classes of DeMorgan lattices, such as binary products of total orders. The main idea behind our reduction technique was inspired by Fitting's work [13] on the connection between a "multi-expert" and "many-valued" semantics of a propositional modal logic. However, in Fitting's work negation is treated as pseudo-complement, which is not appropriate for common multi-valued model checking problems. Our work supports DeMorgan negation as well as other notions of negation. Also, we define reduced models using join-irreducible elements rather than proper prime filters, thereby getting stronger intuition and simpler proofs.

Existing work on the direct model checking of multi-valued logics handles either a smaller class of lattices or logics less expressive than the modal mu-calculus. In [2] an algorithm is defined for CTL over L_3, in [6] an automata-theoretic algorithm is defined for LTL over finite linear orders, and in [7] a BDD-based algorithm is defined for CTL over DeMorgan lattices.

The contribution of this paper is not only in generalizing and improving existing work on multi-valued model checking. We prove basic results about EAA that are interesting independently of this application. For example, in generalizing AA to EAA, the notion of run is extended so that it has an associated value. The question arises whether the set of values of all the accepting runs has a maximal element. This question is answered positively in this paper.

In the following section we briefly cover some lattice theory and the modal mu-calculus. In Section 3, we define our reduction method. In Section 4 we define extended alternating automata, and in Section 5 we show how to directly model check with them. We conclude in Section 6 by comparing the reduction and direct approaches to multi-valued model checking.

2 Background

2.1 Lattices and Negation

Consider the interpretation of boolean connectives in basic propositional logic. If the truth constants *true* and *false* are ordered by *true* > *false*, then the standard meaning of conjunction can be expressed as the minimum of its two arguments, and similarly for disjunction and maximum. Alternatively, conjunction can be understood as meet in the sense of lattice theory, and disjunction as join. It is then obvious to see how to generalize conjunction and disjunction to logics in which the truth values are the elements of a lattice. For example, consider lattice $L3$ of Figure 1. The top element 1 represent truth, the bottom element 0 represents falsity, and the middle element $1/2$ represents "unknown whether true or false". Interpreting conjunction as meet, we get for example that $0 \wedge 1/2 = 0$ and $1 \wedge 1/2 = 1/2$.

In this paper we consider the multi-valued interpretation of temporal logic. In a two-valued temporal logic, interpreting a formula relative to a Kripke structure yields a set of states – intuitively the states of the Kripke structure that satisfy the formula. In our multi-valued interpretation, interpreting a formula yields a mapping from a state to a lattice element. The value that a state is mapped to represents the degree to which the state satisfies the formula. Before presenting a multi-valued temporal logic, we fix the class of lattices that will be used.

Let $(A, \leq)$ be an ordered set, and let P be a subset of A. An element x of P is an *upper bound* of P if $x \geq y$ for every element y in P. If the set of upper bounds of P has a least element, it is called the *least upper bound. Lower bound* and *greatest lower bound* are defined dually. We write $x \vee y$ for the least upper bound of $\{x,y\}$, if it exists, and similarly $x \wedge y$ for the greatest lower bound of $\{x,y\}$. We also write $\bigvee P$ for the least upper bound of P, and $\bigwedge P$ for the greatest lower bound of P, again assuming they exist.

An ordered set $(A, \leq)$ is a *lattice* if $x \vee y$ and $x \wedge y$ exist for all x, y in A, and is a *complete* lattice if $\bigvee P$ and $\bigwedge P$ exist for all subsets P of A. A lattice is *distributive* if $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$ for all elements x, y, z of the lattice. A *finite lattice* has only finitely many elements; every such lattice has a greatest element, called *top*, and a least element, called *bottom* (and written $\perp$). Note that every finite lattice is complete, and that every complete lattice has top and bottom elements.

We stated above that conjunction and disjunction can be interpreted as meet and join. However, we also want to work with logics having a negation operator, so we need to use lattices in which a sensible notion of negation can be defined. We now consider several classes of lattices and the notion of negation they support.

Boolean lattices support a strong sense of complement. Every element x in such a lattice has a unique complement $\neg x$ such that $x \vee \neg x$ equals the top element of the lattice and $x \wedge \neg x$ equals the bottom element of the lattice. Lattice L_2 of Fig. 1 is boolean. However, there are "few" boolean lattices – each is isomorphic to a lattice formed from a set A by taking all subsets of A as elements and ordering by set inclusion.

In a *DeMorgan* (or *quasi-boolean*) lattice [1], every element x has a unique complement $\neg x$ such that $\neg\neg x = x$, DeMorgan's laws hold, and $x \leq y$ implies $\neg y \leq \neg x$. DeMorgan lattices can be characterized as lattices with horizontal symmetry [7]. Lattice L_3 of Fig. 1 is DeMorgan, but not boolean. Using DeMorgan complement we get that $\neg 0 = 1$, $\neg 1/2 = 1/2$, and $\neg 1 = 0$

A *Heyting algebra* is a lattice in which every element x has a unique *relative pseudo-complement* $\neg x$ defined as the greatest value y such that $x \wedge y$ equals the bottom element of the lattice. In the case of finite lattices, Heyting algebras and distributive lattices are the same thing [12]. The right-most lattice in Fig. 1 is a Heyting algebra but is not DeMorgan. In this lattice, if we use relative pseudo-complement as complement, we get $\neg a = e$ and $\neg b = a$. In lattice L_3 we get $\neg 0 = 1$, $\neg 1/2 = 0$, and $\neg 1 = 0$. Some DeMorgan lattices are not Heyting algebras. If an extra element b is added to $L_{2,2}$ of Fig. 1 such that b is less than

$(1,1)$, greater than $(0,0)$, and incomparable to $(0,1)$ and $(1,0)$, then one obtains such a lattice.

Reasoning about partial information with three-valued logic based on $L3$ is an important application of multi-valued model checking, and since in this application we want to interpret negation in the DeMorgan sense, we adopt DeMorgan lattices for multi-valued model checking.

2.2 The Modal Mu-Calculus

The modal mu-calculus [17] (also known as the propositional μ-calculus) is a propositional modal logic extended with fixed-point operators. The logic is very expressive and includes as fragments linear-time temporal logic (LTL) [20] and computation-tree logic (CTL) [10].

Without loss of generality, we use a positive form of the modal mu-calculus, in which negation applies only to atomic propositions. Formulas of the logic have the following abstract syntax, where p ranges over a set P of atomic propositions and X ranges over a set Var of fixed-point variables: $\phi ::= p \mid \neg p \mid \phi_1 \wedge \phi_2 \mid \phi_1 \vee \phi_2 \mid \Box\,\phi \mid \Diamond\,\phi \mid X \mid \nu X.\phi \mid \mu X.\phi$. In fixed-point formulas $\nu X.\phi$ and $\mu X.\phi$ the operators ν and μ bind free occurrences of X in ϕ. We refer to this logic as μL.

Recall that a Kripke structure $M = (S, s_0, \Theta, \mathcal{R})$ is a tuple in which S is a set of states, s_0 in S is the initial state, Θ maps a state to a subset of P (i.e. the propositions that hold at the state), and $\mathcal{R} \subseteq S \times S$ is a transition relation assumed to be total. We write $s \to s'$ if $(s, s') \in \mathcal{R}$ and write $succ_{\mathcal{R}}(s)$ for the set $\{s' \in S \mid s \to s'\}$. For a finite set $D \subset I\!N$ we say that M *has degrees in* D if, for every state s of S, the value $|succ_{\mathcal{R}}(s)|$ is in D.

A Kripke structure $M = (S, s_0, \Theta, \mathcal{R})$ over a lattice L differs from a standard Kripke structure in that now Θ maps a state to a mapping from propositions to elements of L. In what follows, $P \to L$ denotes the set of all possible mappings from P to L.

A *valuation* $\mathcal{V}$ over a lattice L maps a variable to a mapping from states to elements of L. We write $()$ for the valuation such that $()(X)(s) = \bot$ for all X and s (it is required here that L has a bottom element), and write $\mathcal{V}[X := f]$ for the valuation that is like $\mathcal{V}$ except that it maps X to f.

We define the meaning $\|M, \phi\|_{\mathcal{V}}$ of a μL formula relative to a Kripke structure $M = (S, s_0, \Theta, \mathcal{R})$ over lattice L as a mapping from S to L. In the following definition the function $f : (S \to L) \to (S \to L)$ is defined by $f(g) = \|M, \phi\|_{\mathcal{V}[X:=g]}$, and νf and μf stand for the greatest and least fixed-points of f. We know f has greatest and least fixed-points by the Knaster-Tarski fixpoint theorem [23] because functions in $S \to L$ ordered by $g_1 \sqsubseteq g_2 = \forall s. g_1(s) \leq g_2(s)$ form a complete lattice with meet and join operators defined as $(g_1 \wedge g_2)(s) = g_1(s) \wedge g_2(s)$ and $(g_1 \vee g_2)(s) = g_1(s) \vee g_2(s)$. Function f preserves $\sqsubseteq$.

Definition 1. *The interpretation $\|M, \phi\|_{\mathcal{V}}$ of a μL formula relative to Kripke structure $M = (S, s_0, \Theta, \mathcal{R})$ and valuation $\mathcal{V}$ over DeMorgan lattice L is defined* *as follows:*

$$
\begin{aligned}
||M, p||_{\mathcal{V}} &= \lambda s.\Theta(s)(p) \\
||M, \neg p||_{\mathcal{V}} &= \lambda s.\neg\Theta(s)(p) \\
||M, \phi_1 \wedge \phi_2||_{\mathcal{V}} &= \lambda s.||M, \phi_1||_{\mathcal{V}}(s) \wedge ||M, \phi_2||_{\mathcal{V}}(s) \\
||M, \phi_1 \vee \phi_2||_{\mathcal{V}} &= \lambda s.||M, \phi_1||_{\mathcal{V}}(s) \vee ||M, \phi_2||_{\mathcal{V}}(s) \\
||M, \Box\, \phi||_{\mathcal{V}} &= \lambda s. \bigwedge \{||M, \phi||_{\mathcal{V}}(s') \mid s \to s'\} \\
||M, \Diamond\, \phi||_{\mathcal{V}} &= \lambda s. \bigvee \{||M, \phi||_{\mathcal{V}}(s') \mid s \to s'\} \\
||M, X||_{\mathcal{V}} &= \mathcal{V}(X) \\
||M, \nu X.\phi||_{\mathcal{V}} &= \nu f \\
||M, \mu X.\phi||_{\mathcal{V}} &= \mu f
\end{aligned}
$$

If ϕ is a closed formula then we write $[(M, s), \phi]$ for the value $||M, \phi||_{()}(s)$ of formula ϕ at state s of Kripke structure M. Given ϕ, (M, s) and L, computing $[(M, s), \phi]$ is called the *multi-valued model-checking problem.* If M is a Kripke structure over lattice L_2, then we write $(M, s) \models \phi$ if $[(M, s), \phi] =$ *true.*

Proposition 1. *The μL semantics of Def. 1 collapses to the standard two-valued semantics of μL when lattice L is L_2 of Fig. 1.*

3 Reduction to 2-Valued Model Checking

In this section we show how multi-valued model checking of a μL formula ϕ relative to a Kripke structure M over an finite distributive lattice L can be performed by model checking ϕ relative to a number of standard Kripke structures. The main idea in this result comes from Fitting's work in [13] for the case of Heyting algebras.

The idea is to use the lattice L to define a set $\mathcal{E}$ of "experts", and then to derive from the multi-valued Kripke structure M a standard Kripke M_e for each expert. From the set of experts e for which M_e satisfies ϕ we then derive the value $[(M, s), \phi]$. The key insight is that for a finite distributive lattice, the experts correspond to the join-irreducible elements of the lattice.

A special handling of negation is required to adapt Fitting's idea to DeMorgan lattices. Let a *dual Kripke structure* $M = (S, s_0, \Theta, \overline{\Theta}, \mathcal{R})$ be a structure for which $(S, s_0, \Theta, \mathcal{R})$ and $(S, s_0, \overline{\Theta}, \mathcal{R})$ are Kripke structures. To interpret a μL formula over a dual Kripke structure, modify the semantic clauses for propositions in Section 2.2 as follows:

$$
\begin{aligned}
||M, p||_{\mathcal{V}} &= \lambda s.\Theta(s)(p) \\
||M, \neg p||_{\mathcal{V}} &= \lambda s.\overline{\Theta}(s)(p)
\end{aligned}
$$

If, given a Kripke structure $M = (S, s_0, \Theta, \mathcal{R})$, we create a dual Kripke structure $M' = (S, s_0, \Theta, \overline{\Theta}, \mathcal{R})$ by defining $\overline{\Theta}(s)(p) = \neg\Theta(s)(p)$, then clearly $||M, \phi||_{\mathcal{V}} = ||M', \phi||_{\mathcal{V}}$ for every μL formula ϕ.

Now we can define the set $\mathcal{E}$ of experts and the Kripke structure M_e of an expert e. Let L be a finite distributive lattice (not necessarily DeMorgan), and let $M = (S, s_0, \Theta, \mathcal{R})$ be a Kripke structure over L. A *join-irreducible element* x of L is an element that is not the bottom element and for which $x = y \vee z$ implies $x = y$ or $x = z$. If L is finite, the join-irreducible elements are easily spotted in the Hasse diagram for L as elements having exactly one lower cover (i.e. there is exactly one line connected to the element from below). The darkened elements in Figure 1 are the join-irreducible ones. The set $\mathcal{E}$ of experts are the join-irreducible elements of L. Borrowing from [13], we say that expert e_1 *dominates* expert e_2 if $e_1 \geq e_2$.

The dual Kripke structure for an expert e is defined as $M_e = (S, s_0, \Theta_e, \overline{\Theta}_e, \mathcal{R})$, where

$$\begin{aligned} \Theta_e(s)(p) &= \Theta(s)(p) \geq e \\ \overline{\Theta}_e(s)(p) &= \neg(\Theta(s)(p)) \geq e \end{aligned}$$

Intuitively, the expert corresponding to a join-irreducible element x in a lattice regards elements of value x or greater as true, and all others as false. Thus $e \geq e'$ means that expert e is less likely to believe something to be true than e'.

Proposition 2. *Let $M = (S, s_0, \Theta, \mathcal{R})$ be a Kripke structure over a finite distributive lattice L, with s in S. Then, $((M_e, s) \models \phi$ and $e \geq e') \Rightarrow (M_{e'}, s) \models \phi$.*

The value of a formula relative a Kripke structure over a lattice L can be determined by checking the standard Kripke structures obtained for each of the experts.

Lemma 1. *Let $M = (S, s_0, \Theta, \mathcal{R})$ be a Kripke structure over a finite distributive lattice L, with s in S, and let x be a join-irreducible element of L. Then, $(M_x, s) \models \phi \Leftrightarrow x \leq [(M, s), \phi]$.*

From this lemma our main theorem follows easily using Birkhoff's representation theorem for finite distributive lattices. Birkhoff's theorem states that a finite lattice is distributive just if it is isomorphic to the lattice on sets in which the elements are down-closed subsets of the join-irreducible elements of the lattice. The following formulation is taken from [11]. We write $\mathcal{J}(L)$ for the set of all join-irreducible elements of a lattice L, and write $\mathcal{O}(A)$ for the lattice having as elements the down-closed subsets of ordered set A, with set inclusion as ordering.

Theorem 1 (Birkhoff's Representation Theorem). *Let L be a finite distributive lattice. Then the map $\eta : L \to \mathcal{O}(\mathcal{J}(L))$ defined by $\eta(a) = \{x \in \mathcal{J}(L) \mid x \leq a\}$ is an isomorphism of L onto $\mathcal{O}(\mathcal{J}(L))$.*

The inverse mapping from $\mathcal{O}(\mathcal{J}(L))$ to L maps a down-closed set $\{x_1, \ldots, x_n\}$ of join-irreducible elements to their join $\bigvee\{x_1, \ldots, x_n\}$. So any element a of a finite distributive lattice can be represented as the join of all the join-irreducible elements less than or equal to a in the lattice.

Theorem 2. *Let $M = (S, s_0, \Theta, \mathcal{R})$ be a Kripke structure over a finite distributive finite lattice L, with s in S. Then, $[(M, s), \phi] = \bigvee\{e \in \mathcal{E} \mid (M_e, s) \models \phi\}$.*

```
A := $\mathcal{E}$;
B := $\emptyset$;
while A $\neq \emptyset$ {
    let $e$ be a maximal element of A;
    if $(M_e, s) \models \phi$ {
        C := $\{e' \in \mathcal{E} | e' \leq e\}$;
        B := B $\cup$ C;
        A := A $-$ C;
    } else {
        A := A $-$ $\{e\}$;
}
return B
```

Fig. 2. A procedure for computing $\{e \in \mathcal{E} \mid (M_e, s) \models \phi\}$

Proof. Proofs are omitted in this extended abstract due to space limitations.

In computing $\bigvee\{e \in \mathcal{E} \mid (M_e, s) \models \phi\}$, if one finds that $(M_{e_1}, s) \models \phi$, and it is the case that $e_1 \geq e_2$, then it must be that $(M_{e_2}, s) \models \phi$, so M_{e_2} need not be checked. Figure 2 is an algorithm for computing $\{e \in \mathcal{E} \mid (M_e, s) \models \phi\}$ that uses this idea. The set A is the set of experts for which checking remains to be done.

Consider the application of this algorithm to model checking over lattice L_3 of Fig. 1. The join-irreducible elements are 1/2 and 1, so our experts are $e_1 = 1$ and $e_2 = 1/2$, where e_1 dominates e_2. The intuition is that e_1 is a pessimist who regards only value 1 as true, while e_2 is an optimist who regards both 1/2 and 1 as true.

Thanks to our general reduction from multi-valued model checking to traditional model checking and since traditional model checking is a special case of multi-valued model checking, we immediately obtain the following complexity bounds for the multi-valued model-checking problem.

Theorem 3. *Let L be a finite distributive DeMorgan lattice with n join-irreducible elements, and let TL denote μL or any of its fragments. Then, the multi-valued model-checking problem for TL with respect to L can be solved in time linear in n. Moreover, the complexity of multi-valued model checking for TL has the same time and space complexity, both in the size of the Kripke structure and of the formula, as traditional two-valued model checking for TL.*

The linear complexity in the number n of join-irreducible elements of the lattice L given in the previous theorem can be improved for specific classes of lattices. For instance, when a lattice L has all its join-irreducible elements linearly ordered, the precedure of Fig. 2 can be modified to perform a binary search (i.e., checking first the join-irreducible element in the middle of the lattice, then checking the join-irreducible element in the middle of the upper or lower half, etc.) instead of a linear search, hence providing a decision procedure for the multi-valued model-checking problem for L with a worst-case time complexity of $O(log(n))$ instead of $O(n)$.

Our reduction theorem generalizes existing work on reducing multi-valued model checking to two-valued model checking. In [3] a reduction is given for three-valued model checking. In [16], reductions are given for three classes of DeMorgan lattices: total orders, binary products of total orders, and the lattice $2 \times 2 + 2$, which can be obtained from the third lattice of Fig. 1 by adding a new top element f that is greater that element e.

In [13], Fitting defines two kinds of models for propositional modal logic. A *many-valued modal model* is like a Kripke structure over a lattice except that transitions are also multi-valued. The lattice must have a bottom element and pseudo-complements must exists. A *multiple-expert modal model* is like a standard Kripke structure except that it includes a set of *experts* and a binary *dominates* relation over experts. The transition relation is parameterized by experts, and satisfies the condition that if a e_1 transition exists and expert e_1 dominates e_2, then a corresponding e_2 transition must also exist. The truth assignment is also parameterized by experts, and satisfies a similar condition.

Fitting shows that one can derive a multi-expert model from a many-valued model such that a formula ϕ holds in the derived model, relative to an expert e, exactly when the value of ϕ in the multi-valued model is an element of the set that was used to define the expert e. The key idea is that the set of experts of the derived multi-expert model can be defined as the proper prime filters of the lattice. Fitting also shows how one can derive a many-valued model from any multi-expert model.

Our work differs from Fitting's in several ways. First, we use μL rather than propositional modal logic. Second, we define our derived two-valued models using join-irreducible elements rather than proper prime filters. This simpler approach works because for finite distributive lattices, a set of lattice elements is a proper prime filter just if it is equal to the up-closed set of a join-irreducible element. Finally, Fitting treats negation as pseudo-complement, while our reduction is parametric with respect to negation, and therefore supports any form of negation that can be defined on a subclass of finite distributive lattices. The price paid for this generality is that we require that M_e have both Θ and an additional second mapping $\overline{\Theta}$ for the interpretation of atomic propositions. We do require that DeMorgan's laws hold in translating a μL formula with negation to one in which negation appears only on atomic propositions.

[14] concerns AC-lattices, which can be thought of as pairs of graph-isomorphic lattices in which the order relation of one is the inverse of the other. Negation in an AC-lattice is captured as two maps, each mapping an element of one lattice to the isomporphic image in the other. The motivation for these lattices is to allow for an analysis of "conflict" between multiple requirements. A notion of expert similar to Fitting's is used. It is shown, for finite models, that for each of the two "modes" captured by the two lattices in an AC-lattice, the set of views for which a modal mu-calculus formula holds is equal to the set obtained by a interpretation of the formula as a view set. The result differs from ours in that it is based on AC-lattices, in its treatment of negation, and in that it relates

view sets rather than lattice elements directly. Also, by using Birkhoff's theorem along with Lemma 1, we obtain a simpler proof.

4 Extended Alternating Automata

In this section, we recall the definition of *Extended Alternating Automata* (EAA) on trees as introduced in [4] and present new results concerning their properties.

4.1 Definition

Alternating automata on trees (e.g., see [18]) generalize traditional nondeterministic tree automata by allowing several successor states of the automaton to go down along the same branch of the input tree. For example, consider an alternating automaton in state s that is reading node x of an input tree such that x is labeled with $T(x)$ and has k successors. The transition function ρ of the automaton can specify that $\rho(s, T(x), k) = ((1, s') \wedge (1, s'')) \vee (2, s')$; this intuitively means that the automaton can *either* split into two copies, one of which will move to state s' and will read next the first successor of x, while the second will move to state s'' and will also read next the first successor of x, *or* the automaton can instead move to state s'' without splitting and read next the second successor of x. Constants *true* and *false* can also appear in an expression mapped to by the transition function.

Extended alternating automata [4] generalize alternating automata by allowing the expressions specified in the transition function to include meet and join operations of a lattice, and any values of the lattice as constants. A standard alternating automaton is thus an EAA based on the lattice $(\{\mathit{true}, \mathit{false}\}, \leq)$.

Semantically, EAA generalize the notion of a run in alternating automata. A run of an alternating automaton on an input tree is itself a tree, with each node of the run labeled by a node of the input tree. The labels on a node of the run and its children must satisfy the automaton's transition function. In an EAA, each node of the run is additionally labeled with a value of the underlying lattice, and the values labeling a node of a run and its children must again satisfy the transition function. Every run itself has a value, which is the value labeling the root node. A run is accepting if it satisfies the acceptance condition of the EAA and also has no node labeled with the bottom element of the lattice. In a standard alternating automaton each node of a run is implicitly labeled with value *true*.

We now formalize the above discussion. A *tree* τ is a subset of $\mathbb{N}^*$ such that if $x \cdot c \in \tau$ then $x \in \tau$ and $x \cdot c' \in \tau$ for all $1 \leq c' < c$. The elements of τ are called its *nodes*, with ϵ called the *root*. Given a node x of τ, values of the form $x \cdot i$ in τ are called the *children* or *successors* of x. The number of successors of x is called the *degree* of x. A node with no successors is called a *leaf*. Given a set $D \subset \mathbb{N}$, a D*-tree* is a tree in which the degree of every node is in D. A Σ*-labeled* tree is a pair (τ, T) in which τ is a tree and $T : \mathbb{N}^* \to \Sigma$ is a labeling function.

Let $L = (B, \wedge, \vee)$ be a complete lattice, and let $\mathcal{B}^+(X)$ stand for the set of terms built from elements in a set X using $\wedge$ and $\vee$. A *tree EAA over L* is a tuple $A = (\Sigma, D, S, s_0, \rho, F)$, where Σ is a nonempty finite alphabet, S is a nonempty finite set of states, $s_0 \in S$ is the initial state, F is an acceptance condition, $D \subset I\!N$ is a finite set of arities, and $\rho : S \times \Sigma \times D \to \mathcal{B}^+((I\!N \times S) \cup B)$ is a transition function, where $\rho(s, a, k) \in \mathcal{B}^+((\{1, \ldots, k\} \times S) \cup B)$ is defined for each s in S, a in Σ, and k in D. Various types of acceptance conditions F can be used with EAA, just as in alternating automata, and are discussed below.

A *v-run* of a tree EAA A on a Σ-labeled leafless D-tree (τ, T) is an $I\!N^* \times S \times B$-labeled tree (τ_σ, T_σ). A node in τ_σ labeled by (x, s, v) describes a copy of automaton A that reads the node x of τ_σ in the state s of A and has value $v \in B$ associated with it. Formally, a v-run (τ_σ, T_σ) is an $I\!N^* \times S \times B$-labeled tree, defined as follows.

- $T_\sigma(\epsilon) = (\epsilon, s_0, v)$
- Let $y \in \tau_\sigma$, $T_\sigma(y) = (x, s, v')$, $arity(x) = k$, and $\rho(s, T(x), k) = \theta$. Then there is a (possibly empty) set $Q = \{(c_1, s_1, v_1), \ldots, (c_n, s_n, v_n)\} \subseteq \{1, \ldots, k\} \times S \times B$ such that
  - for all $1 \leq i, j \leq n$, $c_i = c_j$ and $s_i = s_j$ implies $v_i = v_j$,
  - $Eval(Q, \theta) = v'$, and
  - for all $1 \leq i \leq n$, we have $y \cdot i \in \tau_\sigma$ and $T_\sigma(y \cdot i) = (x \cdot c_i, s_i, v_i)$

$Eval(Q, \theta)$ denotes the value of the expression θ obtained by replacing each term (c_i, s_i) in θ by v_i if $(c_i, s_i, v_i) \in Q$ and replacing each term (c_i, s_i) in θ by $\perp$ if $(c_i, s_i, v_i) \notin Q$.

A v-run σ is *accepting* if (1) the value associated with each node of the run is not $\perp$ and (2) all infinite branches of the run satisfy the acceptance condition F. As with traditional alternating automata, various types of acceptance conditions can be used. For instance, a path w satisfies a *parity acceptance condition* $F = \{F_1, F_2, \ldots, F_n\}$ with $F_1 \subseteq F_2 \subseteq \ldots \subseteq F_n$ if the minimal index i for which some state s in F_i appears infinitely often along w is even.

Note that an accepting run can have finite branches: if, for some $y \in \tau_\sigma$, $T_\sigma(y) = (x, s, v)$ and $\rho(s, T(x), arity(x)) = v$ with $v \in L$ and $v \neq \perp$, then y does not need to have any successor.

A tree EAA A accepts a Σ-labeled leafless D-tree (τ, T) with value v if there exists an accepting v-run of A on that tree. We define the language $\mathcal{L}_v(A)$ as follows (for $v \neq \perp$): $\mathcal{L}_v(A) = \{(\tau, T) \mid A \text{ accepts } (\tau, T) \text{ with value } v\}$. For notational convenience, we define $\mathcal{L}_\perp(A)$ as $\{(\tau, T) \mid A \text{ has no accepting run on } (\tau, T)\}$.

When D is a singleton, A runs over trees with a fixed branching degree. In particular, a *word EAA* is simply a tree EAA in which $D = \{1\}$.

4.2 Properties

We now establish two fundamental properties of EAA. Our first and main result states that, for any EAA and any input tree, there always exists a maximum value v of L for which the EAA has an accepting v-run on the input tree.

Theorem 4 (Maximum-value theorem). *Let A be a tree EAA over a complete lattice L, and let (τ, T) be a Σ-labeled leafless D-tree. Then the subset $\{v \mid (\tau, T) \in \mathcal{L}_v(A)\}$ of L has a unique maximum value, which we denote by $Max(A, (\tau, T))$.*

Note that it is not generally true that if an EAA has an accepting v_1-run and an accepting v_2-run on an input tree then the EAA has an accepting $(v_1 \vee v_2)$-run on the input tree. In what follows, we will simply write $Max(A)$ for $Max(A, \{a^\omega\})$ when A is a word EAA on a 1-letter alphabet $\{a\}$.

Given any EAA $A = (\Sigma, D, S, s_0, \rho, F)$ on a DeMorgan lattice L, its *dual* automaton $\tilde{A} = (\Sigma, D, S, s_0, \tilde{\rho}, \overline{F})$ is defined as the EAA over L obtained by dualizing the transition function ρ of A (i.e., replacing $\wedge$ by $\vee$, $\vee$ by $\wedge$ and every constant c in L by $\neg c$) and by complementing the acceptance condition F of A (i.e., if $\mathcal{F}$ and $\overline{\mathcal{F}}$ denote the subset of S^ω that satisfies the acceptance condition F and $\overline{F}$ respectively, then $\overline{\mathcal{F}} = S^\omega \setminus \mathcal{F}$). Our second result relates an EAA and its dual.

Theorem 5 (Complementation theorem). *Let A be an EAA over a DeMorgan lattice L, let $\tilde{A}$ be its dual EAA, and let (τ, T) be a Σ-labeled leafless D-tree. Then we have $Max(\tilde{A}, (\tau, T)) = \neg Max(A, (\tau, T))$.*

This result generalizes the traditional complementation theorem on AAs [21].

5 Model Checking with EAA

The model-checking procedure for multi-valued logics using EAAs generalizes the automata-theoretic approach to 2-valued model checking with AAs of [18]. Our procedure computes the value $[(M, s), \phi]$ defined by a μL formula ϕ evaluated in state s of a Kripke structure M over a DeMorgan lattice L. First, we translate ϕ to an EAA A_ϕ. Then we build a product automaton from A_ϕ and M in such a way that the maximum value that labels an accepting run of the product automaton is $[(M, s), \phi]$. We now present these steps in detail.

We begin with a translation of μL formulas to EAAs. The property we want of the translation is that the value of the maximum accepting run of the EAA for formula ϕ and an input tree (τ, T) agrees with the value $[(\tau, T), \phi]$ defined by the semantics of μL (with (τ, T) viewed as a Kripke structure over L).

Definition 2. *Let ϕ denote a μL formula and let L denote a DeMorgan lattice. Let $cl(\phi)$ denote the set of all subformulas of ϕ, and let P be the set of atomic propositions appearing in $cl(\phi)$. Given a set $D \subset I\!N$, we define a parity EAA $A_{D,\phi} = (P \to L, D, cl(\phi), \phi, \rho, F)$ where the transition function ρ and the parity acceptance condition F are defined as follows:*

- *For all $\sigma : P \to L$ and $k \in D$, we define:*
  - $\rho(p, \sigma, k) = \sigma(p)$.
  - $\rho(\neg p, \sigma, k) = \neg\sigma(p)$.
  - $\rho(\phi_1 \wedge \phi_2, \sigma, k) = split(\rho(\phi_1, \sigma, k) \wedge \rho(\phi_2, \sigma, k))$.
  - $\rho(\phi_1 \vee \phi_2, \sigma, k) = split(\rho(\phi_1, \sigma, k) \vee \rho(\phi_2, \sigma, k))$.

  - $\rho(\Box\,\phi, \sigma, k) = split(\bigwedge_{c=1}^{k}(c, \phi))$.
  - $\rho(\Diamond\,\phi, \sigma, k) = split(\bigvee_{c=1}^{k}(c, \phi))$.
  - $\rho(\nu X.f(X), \sigma, k) = split(\rho(f(\nu X.f(X)), \sigma, k))$.
  - $\rho(\mu X.f(X), \sigma, k) = split(\rho(f(\mu X.f(X)), \sigma, k))$.

  where the function split $: \mathcal{B}^+(I\!N \times cl(\phi)) \to \mathcal{B}^+(I\!N \times cl(\phi))$ *is defined recursively by* $split(v) = v$ *for* $v \in L$, $split(\phi_1 \wedge \phi_2) = split(\phi_1) \wedge split(\phi_2)$, $split(\phi_1 \vee \phi_2) = split(\phi_1) \vee split(\phi_2)$, $split((c, \phi_1 \wedge \phi_2)) = split((c, \phi_1)) \wedge split((c, \phi_2))$, $split((c, \phi_1 \vee \phi_2)) = split((c, \phi_1)) \vee split((c, \phi_2))$, *and* $split((c, \phi)) = (c, \phi)$ *when* ϕ *is of the form* p, $\neg p$, $\Box\,\phi'$, $\Diamond\,\phi'$, $\nu X.f(X)$ *or* $\mu X.f(X)$.
- F *is defined exactly as in [18]: if* $d \geq 1$ *denotes the maximal alternation level of subformulas of* ϕ, G_i *denotes the set of all the* ν*-formulas in* $cl(\phi)$ *of alternation level* $i \leq d$, *and* B_i *denotes the set of all the* μ*-formulas in* $cl(\phi)$ *of alternation level* $i \leq d$, *then* $F = \{F_1, F_2, \ldots, F_{2d}\}$ *where* $F_0 = \emptyset$ *and for all* $1 \leq i \leq d$, $F_{2i-1} = F_{2i-2} \cup B_i$ *and* $F_{2i} = F_{2i-1} \cup G_i$.

This translation is nearly identical to the translation from μL to alternating automata given in [18]. The only difference is the translation of formulas of the form p and $\neg p$ where σ represents in our case a mapping from atomic propositions to elements in the lattice L. (Definition 2 is included here to make our paper self-contained.) The correctness of this translation is established by the following theorem.

Theorem 6. *Let* ϕ *denote a* μL *formula and let* L *denote a DeMorgan lattice. Then a parity EAA* $A_{D,\phi}$ *for* ϕ *can be constructed in linear time such that* $[((\tau, T), \epsilon), \phi)] = Max(A_{D,\phi}, (\tau, T))$ *for every leafless* D*-tree* (τ, T) *on* L.

In the next step of the procedure, we compute the product of a Kripke structure and an EAA representing a μL formula. The product construction defined here is again nearly identical to that given for alternating automata in [18].

Definition 3. *Let* ϕ *be a* μL *formula over a DeMorgan lattice* L, $M = (S, s_0, \Theta, \mathcal{R})$ *be a Kripke structure over* L *with degrees in* D, *and* $A_{D,\phi} = (P \to L, D, Q_\phi, q_0, \rho_\phi, F)$ *be a parity EAA representing* ϕ. *Then the* product automaton $A_{M,\phi} = (\{a\}, S \times Q_\phi, (s_0, q_0), \rho, F)$ *of* M *and* $A_{D,\phi}$ *is a parity word EAA over a 1-letter alphabet with at most* $O(|S| \cdot |Q_\phi|)$ *states, where* ρ *and* F *are defined as follows:*

- *For all* $q \in Q_\phi$, $s \in S$, *if* $succ_{\mathcal{R}}(s) = (s_1, \ldots, s_n)$ *and* $\rho_\phi(q, \Theta(s), n) = \theta$, *then* $\rho((s, q), a) = \theta'$ *where* θ' *is obtained from* θ *by replacing each atom* (c, q') *in* θ *by* (s_c, q').
- *If* $F_\phi = \{F_1, F_2, \ldots, F_m\}$ *is a parity acceptance condition, then so is* $F = \{(S \times F_1), (S \times F_2), \ldots, (S \times F_m)\}$.

The product automaton $A_{M,\phi}$ is used to prove the following.

Theorem 7. *Let* ϕ *be a* μL *formula,* $M = (S, s_0, \Theta, \mathcal{R})$ *be a Kripke structure over a DeMorgan lattice* L *with degrees in* D, *and* s *be a state in* S. *Then, there exists a parity word EAA* $A_{M,\phi}$ *over a 1-letter alphabet such that* $[(M, s), \phi)] = Max(A_{M,\phi})$.

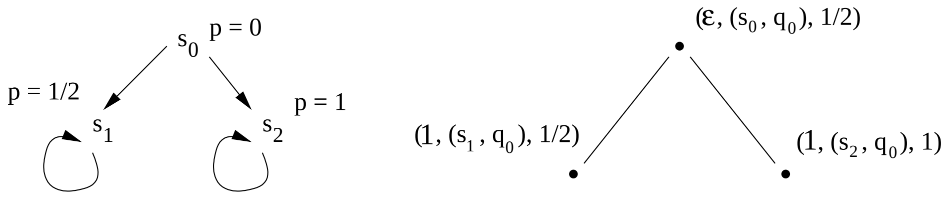

Fig. 3. Example Kripke structure M and accepting run

In the final step of the model-checking procedure, we compute the value $Max(A_{M,\phi})$ of the product EAA.

Theorem 8. *Given a parity word EAA $A_{M,\phi}$ over L with a 1-letter alphabet, computing $Max(A_{M,\phi})$ has the same complexity as checking whether the language accepted by a parity word AA with a 1-letter alphabet is nonempty, i.e., can be done in nondeterministic polynomial time.*

Algorithms for computing $Max(A)$ of a word EAA A over a 1-letter alphabet are similar to algorithms for checking emptiness of AAs over a 1-letter alphabet. The only difference is that the algorithms dealing with EAAs propagates values in L instead of values in the set $\{true, false\}$ as with traditional AAs. Since the best known upper bound for the traditional (2-valued) μL model-checking problem is that it is in NP∩co-NP, this bound carries over to the multi-valued case. However, computing $Max(A_{M,\phi})$ can be done more efficiently for some subclasses of μL. For instance, the EAA corresponding to a CTL formula is *weak* [18], and computing the value $Max(A_{M,\phi})$ of the product of a weak EAA with a Kripke structure can be done in linear time [4].

Example 1. Consider the μL formula $\mu X.p \vee \Box X$, which is equivalent to the CTL formula AFp. By applying the construction of Definition 2, we obtain a tree EAA with a single state q_0, an acceptance condition $F = \emptyset$ and the following transition function:

$$\rho(q_0, \sigma, k) = \sigma(p) \vee \bigwedge_{c=1}^{k} (c, q_0)$$

We next take the product of this automaton with the Kripke structure M over L_3 shown on the left of Figure 3. The figure is labeled to show the value of the atomic proposition p at each state. Using the product construction of Definition 3, we obtain a (weak) word EAA over a 1-letter alphabet with no accepting states and the following transition function:

$$\begin{aligned}
\rho((s_0, q_0), a, 1) &= 0 \vee ((s_1, q_0) \wedge (s_2, q_0)) \\
\rho((s_1, q_0), a, 1) &= 1/2 \vee (s_1, q_0) \\
\rho((s_2, q_0), a, 1) &= 1 \vee (s_2, q_0)
\end{aligned}$$

This EAA has an accepting 1/2-run, which is shown on the right in Figure 3. The value 1/2 is the greatest value v for which there is an accepting v-run, so by Theorem 7, we know that $[(M, s_0), \mu X.p \vee \Box X] = 1/2$.

6 Discussion

It was mentioned in the introduction that, depending on the circumstances, multi-valued model checking may best be done directly or by reduction. Having defined each approach we now reconsider this question.

One issue is whether the lattice being used is infinite. This issue might seem to favor the direct approach, since reduction would require infinitely many two-valued model checks. However, in a finite-state Kripke structure with finitely-many different atomic propositions, at most finitely-many lattice elements will appear. From these, by closing under meet and join, one obtains a finite sublattice of the original lattice. This finite lattice can be used in place of the original one for multi-valued model checking. This idea can be used independently of whether model checking is to be performed directly or by reduction. Also, the idea can be used to attempt to reduce the size of a finite lattice.

More important is the degree to which the two approaches can work in an "on-the-fly" fashion, computing whatever information is necessary to solve the problem at hand on a demand-driven basis. The multi-valued model checking problem has as inputs a lattice, a Kripke structure, and a temporal-logic formula. In the reduction approach, only the lattice and Kripke structure are used in building the two-valued Kripke structures, each of which can then be model checked possibly on-the-fly, thus using the formula to guide the verification needs. In contrast, the direct approach can make use of all three inputs together to further limit computational resources. For instance, consider a lattice of n incomparable elements plus a top and bottom element, and suppose the formula we wish to model check is simply the atomic proposition p. In the reduction approach we must then perform n model checks. In the direct approach we will perform a single model check that examines only the initial state of the multi-valued Kripke structure and reads only the value of p, which requires reading only $log(n)$ bits.

References

1. L. Bolc and P. Borowik. *Many-Valued Logics.* Springer Verlag, 1992.
2. G. Bruns and P. Godefroid. Model Checking Partial State Spaces with 3-Valued Temporal Logics. In *Proceedings of the 11th Conference on Computer Aided Verification*, volume 1633 of *Lecture Notes in Computer Science*, pages 274–287, Trento, July 1999. Springer-Verlag.
3. G. Bruns and P. Godefroid. Generalized Model Checking: Reasoning about Partial State Spaces. In *Proceedings of CONCUR'2000 (11th International Conference on Concurrency Theory)*, volume 1877 of *Lecture Notes in Computer Science*, pages 168–182, University Park, August 2000. Springer-Verlag.
4. G. Bruns and P. Godefroid. Temporal Logic Query Checking. In *Proceedings of LICS'2001 (16th IEEE Symposium on Logic in Computer Science)*, pages 409–417, Boston, June 2001.
5. W. Chan. Temporal-Logic Queries. In *Proceedings of the 12th Conference on Computer Aided Verification*, volume 1855 of *Lecture Notes in Computer Science*, pages 450–463, Chicago, July 2000. Springer-Verlag.

6. M. Chechik, B. Devereux, and A.Gurfinkel. Model-Checking Infinite State-Space Systems with Fine-Grained Abstractions Using SPIN. In *Proceedings of SPIN Workshop on Model-Checking Software*, 2001.
7. M. Chechik, B. Devereux, S. Easterbrook, and A. Gurfinkel. Multi-valued symbolic model checking. Technical Report 448, Computer Systems Research Group, University of Toronto, 2001.
8. M. Chechik and W. Easterbrook. A framework for multi-valued reasoning over inconsistent viewpoints. In *Proceedings of International Conference on Software Engineering (ICSE'01)*, May 2001.
9. M. Chechik, W. Easterbrook, and A. Gurfinkel. Model exploration with temporal logic query checking. In *Proceedings of FSE '02*, November 2002.
10. E. M. Clarke and E. A. Emerson. Design and Synthesis of Synchronization Skeletons using Branching-Time Temporal Logic. In D. Kozen, editor, *Proceedings of the Workshop on Logic of Programs,* Yorktown Heights, volume 131 of *Lecture Notes in Computer Science*, pages 52–71. Springer-Verlag, 1981.
11. B.A. Davey and H.A. Priestly. *Introduction to Lattices and Order.* Cambridge University Press, 1990.
12. M. Fitting. Many-Valued Modal Logics I. *Fundamenta Informaticae*, 15:235–254, 1992.
13. M. Fitting. Many-Valued Modal Logics II. *Fundamenta Informaticae*, 17:55–73, 1992.
14. M. Huth and S. Pradham. Lifting assertion and consistency checkers from single to multiple viewpoints. Technical report, Dept. of Computing, Imperial College, London, 2002.
15. Stephen Cole Kleene. *Introduction to Metamathematics.* North Holland, 1987.
16. B. Konikowska and W. Penczek. Reducing model checking from multi-valued CTL* to CTL*. In *Proceedings of CONCUR 2002, LNCS 2421*, 2002.
17. D. Kozen. Results on the Propositional Mu-Calculus. *Theoretical Computer Science*, 27:333–354, 1983.
18. O. Kupferman, M. Y. Vardi, and P. Wolper. An Automata-Theoretic Approach to Branching-Time Model Checking. *Journal of the ACM*, 47(2):312–360, March 2000.
19. Z. Lotfi. Fuzzy Sets. *Information and Control*, 8:338–353, 1965.
20. Z. Manna and A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems: Specification.* Springer-Verlag, 1992.
21. D. E. Muller and P. E. Schupp. Simulating alternating tree automata by nondeterministic automata: New results and new proofs of the theorems by Rabin, McNaughton and Safra. *Theoretical Computer Science*, 141(1–2):69–108, 1995.
22. K. Segerberg. Some Modal Logics Based on a Three-Valued Logic. *Theoria*, 33:53–71, 1967.
23. A. Tarski. A lattice-theoretical fixpoint theorem and its applications. *Pacific J. of Maths*, 5:285–309, 1955.