Hybrid Logics:
The Search for Decidability and Tractability Frontiers

Moshe Y. Vardi

Rice University

Modal Logic

- **Classical logic**: A proposition is either true or false
- **Modal logic**: *modes* of truth
  - $\Box p$: p is *necessarily* true
  - $\Diamond p$: p is *possibly* true
  - $\Box p = \neg\Diamond\neg p$

Possible-World Semantics: $K = (W, R, L)$

- W: set of *worlds* or (*states*)
- $R \subseteq W^2$: *possibility* relation
- $L : W \rightarrow 2^{Prop}$: labeling

Semantics:

- $K, w \models p$ if $p \in L(w)$
- $K, w \models \Box\varphi$ if $K, v \models \varphi$ whenever $(w, v) \in R$
- $K, w \models \Diamond\varphi$ if $K, v \models \varphi$ for some $(w, v) \in R$

The Modal Revolution

Realization: $\Box p$ can mean whatever you want it to mean!

- p is *known*: epistemic logic
- p is *believed*: doxastic logic
- p is *mandatory*: deontic logic
- p is *provable*: alethic logic

Proliferation:

- Allow $\Box_1, \Box_2, \ldots$
- with possibility relations $R_1, R_2, \ldots$

Description Logic: [Schild, 1991]

- Consider W as a set of entities
- Propositions are sets of entities
- Relations are *roles* – e.g., xParent$y$
- $[Parent]Girl$: has parents with girls only

More Descriptive Power

Role Reversal: [Pratt, 1976]

- $xRy \Rightarrow yR^{-}x$

Example: $[Parents^{-}]Working$ – all parents are working

Example: $\langle Parents^{-}\rangle\langle Parent\rangle female$ – has a female immediate sibling.

Graded modalities: [Goble, 1970, Hollunder&Baader, 1991]

- $\langle Parent\rangle_{\geq 2}Girl$: has at last two girls
- $[Parents^{-}]_{\leq 1}Working$: has at most one working parent

Number Encoding: unary vs binary [Tobies, 2001]

Adding Fixpoints

More expressive power needed:

- “Has a female sibling”, where “sibling” is transitive closure of immediate sibling.
- Cyclic definitions

Solution: *fixpoints* (μ-calculus) [Kozen, 1983]

- *Intuitively*:
  - $\mu X.\varphi$: Smallest set of objects satisfying φ
  - $\nu X.\varphi$: Largest set of objects satisfying φ

Challenging Formalism!

- $\nu X.\mu Y.(X \wedge (p \vee \diamond Y))$

Nominals

Nominal = Name [Prior, 1967, Bull, 1970, Blackburn, 1993, Gargov&Goranko, 1993]

- E.g., *Moshe*
- Formally, a *nominal* is a proposition that denotes a singleton set of entities.
- $\langle Parent \rangle Moshe$: Parent of Moshe

Hybrid Logics: logics with nominals [Blackburn, 2000]

Reasoning

Satisfiability of φ: $K, w \models \varphi$ for some K, w

Important: Most reasoning tasks to description logic reduce to satisfiability checking.

Bad News: μ-calculus with (1) nominals, (2) inverse roles, and (3) graded modalities is undecidable [Bonatti&Peron, 2004]

Good News: μ-calculus with each two of the above extensions is *decidable in EXPTIME*.

Sharp Undecidability Result

Functional Roles: role R with functional relations. i.e., single successor:

- $\langle R \rangle_{\geq 1} \mathbf{true}$ and $[R]_{\leq 1} \mathbf{true}$ must hold.

Theorem: μ-calculus with (1) nominals, (2) inverse roles, and (3) functional roles is undecidable [Bonatti&Peron, 2004]

Proof technique:

- Encode infinite grid.

- Encode infinite tiling problem.

Rabin Automata on Infinite k-ary Trees

Labeled Infinite k-ary Tree: $\tau : \{0, \ldots, k-1\}^* \to \Sigma$

Rabin Automaton: $A = (\Sigma, S, S_0, \rho, \alpha)$

- Σ: finite alphabet
- S: finite state set
- $S_0 \subseteq S$: initial state set
- ρ: transition function
  - $\rho \ : \ S \times \Sigma \to 2^{S^k}$
- α: acceptance condition
  - $\alpha = \{(G_1, B_1), \ldots, (G_l, B_l)\}$, $G_i, B_i \subseteq S$
  - *Acceptance*: along every branch, for some $(G_i, B_i) \in \alpha$, G_i is visited infinitely often, and B_i is visited finitely often.

Emptiness of Tree Automata

Nonemptiness: $L(A) \neq \emptyset$

Nonemptiness of Automata on Finite Trees: PTIME test (Doner, 1965)

Emptiness of Automata on Infinite Trees: Difficult

- Rabin, 1969: non-elementary
- Hossley+Rackoff, 1972: 2EXPTIME
- Rabin, 1972: EXPTIME
- Emerson, V.+Stockmeyer, 1985: In NP
- Emerson+Jutla, 1991: NP-complete

Logic and Automata for Infinite Trees

Monadic Second-Order Logic (MSO) for Trees:

- Unary predicates: $P_a(x)$, for $a \in \Sigma$
- Binary predicates: $E_1(x,y), \ldots, E_k(x,y)$

Quantification:

- *First order*: quantification over nodes
- *Second order*: quantification over sets of nodes

Theorem [Rabin, 1969]:
Tree MSO $\equiv$ Tree Automata

Corollary: Decidability of tree MSO on – one of the most powerful decidability results in logic.

Standard technique in 1970s: Prove decidability via reduction to MSO on trees.

- Can be applied to many hybrid logics
- Requires the tree-model property
- *Nonelementary complexity*
- Can we get elementary automata-theoretic procedures?

Nondeterminism in Complexity Theory

Intuition: "It is easier to criticize than to do."

P vs NP:

PTIME: Can be *solved* in polynomial time

NPTIME: Can be *checked* in polynomial time

Complexity Hierarchy:

LOGSPACE $\subseteq$ NLOGSPACE $\subseteq$ PTIME $\subseteq$ NPTIME
$\subseteq$ PSPACE $=$ NPSPACE $\subseteq$ EXPTIME $\subseteq$ NEXPTIME $\subseteq$ $\ldots$

Co-Nondeterminism

Intuition:

- *Nondeterminism*: check *solutions* – e.g., *satisfiability*
- *Co-nondeterminism*: check *counterexamples* – e.g., *unsatisfiablity*

Complexity Hierarchy:

$$\begin{array}{ccc}
 & \text{LOGSPACE} & \\
 & \subseteq & \\
\text{NLOGSPACE} & = & \text{co-NLOGSPACE} \\
 & \subseteq & \\
 & \text{PTIME} & \\
 & \subseteq & \\
\text{NPTIME} & & \text{co-NPTIME} \\
 & \parallel & \\
 & \text{PSPACE} & \\
\text{NPSPACE} & = & \text{co-NPSPACE} \\
 & \subseteq & \\
 & \text{EXPTIME} & \\
 & \vdots &
\end{array}$$

Alternation

(Co)-Nondeterminism–Perspective Change:

- *Old*: Checking (solutions or counterexamples)
- *New*: Guessing moves
  - *Nondeterminism*: existential choice
  - *Co-Nondeterminism*: universal choice

Alternation: Chandra-Kozen-Stockmeyer, 1981
Combine $\exists$-choice and $\forall$-choice

- $\exists$-state: $\exists$-choice
- $\forall$-state: $\forall$-choice

Easy Observations:

- NPTIME $\subseteq$ APTIME $\supseteq$ co-NPTIME
- APTIME $=$ co-APTIME

Example: Boolean Satisfiability

φ: Boolean formula over $x_1, \ldots, x_n$

Decision Problems:

1. **SAT**: *Is* φ *satisfiable?* – NPTIME

   Guess a truth assignment τ and check that $\tau \models \varphi$.

2. **UNSAT**: *Is* φ *unsatisfiable?* – co-NPTIME

   Guess a truth assignment τ and check that $\tau \models \varphi$.

3. **QBF**: *Is* $\exists x_1 \forall x_2 \exists x_3 \ldots \varphi$ *true?* – APTIME

   Check that for some x_1 for all x_2 for some x_3 $\ldots$ φ holds.

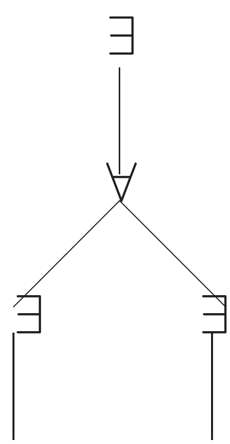
Alternation = Games

Players: $\exists$-player, $\forall$-player

- $\exists$-state: $\exists$-player chooses move
- $\forall$-state: $\forall$-player chooses move

Acceptance: $\exists$-player has a winning strategy

Run: Strategy tree for $\exists$-player



Alternation and Unbounded Parallelism

"**Be fruitful, and multiply**":

- $\exists$-move: fork *disjunctively*
- $\forall$-move: fork *conjunctively*

Note:

- Minimum communication between child processes
- Unbounded number of child processes

Alternating Complexity Classes

Upward Collapse:

- ALOGSPACE=PTIME
- APTIME=PSPACE
- APSPACE=EXPTIME

Applications:

- "In APTIME" $\longrightarrow$ "in PSPACE"
- "APTIME-hard" $\longrightarrow$ "PSPACE-hard".

QBF:

- Natural algorithm is in APTIME $\longrightarrow$ "in PSPACE"
- Prove APTIME-hardness à la Cook $\longrightarrow$ "PSPACE-hard".

Corollary: QBF is PSPACE-complete.

Modal Model Checking

Input:

- φ: modal formula
- $M = (W, R, L)$: Kripke structure
- $w \in W$: world

Problem: $M, w \models \varphi$?

Algorithm: K-$MC(\varphi, M, w)$

case
φ propositional: return $L(w) \models \varphi$
$\varphi = \theta_1 \vee \theta_2$: ($\exists$-branch) return K-$MC(\theta_i, M, w)$
$\varphi = \theta_1 \wedge \theta_2$: ($\forall$-branch) return K-$MC(\theta_i, M, w)$
$\varphi = \Diamond\psi$: ($\exists$-branch) return K-$MC(\psi, M, u)$ for $u \in R(w)$
$\varphi = \Box\psi$: ($\forall$-branch) return K-$MC(\psi, M, u)$ for $u \in R(w)$
esac.

Correctness: Immediate!

Complexity Analysis

Algorithm's state: (θ, M, u)

- θ: $O(\log|\varphi|)$ bits
- M: fixed
- u: $O(\log|M|)$ bits

Conclusion: ASPACE$[\log|M| + \log|\varphi|]$

Therefore: K-MC $\in$ ALOGSPACE=PTIME (Clarke&Emerson, 1981).

Modal Satisfiability

- $sub(\varphi)$: all subformulas of φ

- *Valuation* for φ – α: $sub(\varphi) \rightarrow \{0, 1\}$

  Propositional consistency:

  - $\alpha(\varphi) = 1$
  - Not: $\alpha(p) = 1$ and $\alpha(\neg p) = 1$
  - Not: $\alpha(p) = 0$ and $\alpha(\neg p) = 0$
  - $\alpha(\theta_1 \wedge \theta_2) = 1$ implies $\alpha(\theta_1) = 1$ and $\alpha(\theta_2) = 1$
  - $\alpha(\theta_1 \wedge \theta_2) = 0$ implies $\alpha(\theta_1) = 0$ or $\alpha(\theta_2) = 0$
  - $\alpha(\theta_1 \vee \theta_2) = 1$ implies $\alpha(\theta_1) = 1$ or $\alpha(\theta_2) = 1$
  - $\alpha(\theta_1 \vee \theta_2) = 0$ implies $\alpha(\theta_1) = 0$ and $\alpha(\theta_2) = 0$

Definition: $\Box(\alpha) = \{\theta : \alpha(\Box\theta) = 1\}$.

Lemma: φ is satisfiable iff there is a valuation α for φ such that if $\alpha(\Diamond\psi) = 1$, then $\psi \wedge \bigwedge \Box(\alpha)$ is satisfiable.

Intuition

Lemma: φ is satisfiable iff there is a valuation α for φ such that if $\alpha(\Diamond\psi) = 1$, then $\psi \wedge \bigwedge \Box(\alpha)$ is satisfiable.

Only if: $M, w \models \varphi$
Take: $\alpha(\theta) = 1 \leftrightarrow M, w \models \theta$

If: Satisfy each $\Diamond$ separately

$$\begin{tikzcd} & \Box\beta, \Box\gamma, \Diamond\delta, \Diamond\eta \arrow[dl] \arrow[dr] & \\ \beta, \gamma, \delta & & \beta, \gamma, \eta \end{tikzcd}$$

Algorithm

Algorithm: $K\text{-}SAT(\varphi)$

($\exists$-branch): Select valuation α for φ
($\forall$-branch): Select ψ such that $\alpha(\Diamond\psi) = 1$, and return $K\text{-}SAT(\psi \wedge \bigwedge \Box(\alpha))$

Correctness: Immediate!

Complexity Analysis:

- Each step is in PTIME.
- Number of steps is polynomial.

Therefore: K-SAT $\in$ APTIME=PSPACE (Ladner, 1977).

In practice: Basis for practical algorithm – valuations selected using a SAT solver.

LTL Refresher

Syntax:

- Propositional logic
- $next\ \varphi$, $\varphi\ until\ \psi$

Temporal structure: $M = (W, R, L)$

- W: worlds
- $R : W \rightarrow W$: successor function
- $L : W \rightarrow 2^{Prop}$: truth assignments

Semantics

- $M, w \models p$ if $p \in \pi(w)$
- $M, w \models next\ \varphi$ if $M, R(w) \models \varphi$

- $M, w \models \varphi\ until\ \psi$ if w $\underset{\varphi}{\bullet} \longrightarrow \underset{\varphi}{\bullet} \longrightarrow \quad \underset{\varphi}{\bullet} \longrightarrow \underset{\psi}{\bullet} \longrightarrow \bullet \ldots$

Fact: $(\varphi\ until\ \psi) \equiv (\psi \vee (\varphi \wedge next(\varphi\ until\ \psi)))$.

Temporal Model Checking

Input:

- φ: temporal formula
- $M = (W, R, \pi)$: temporal structure
- $w \in W$: world

Problem: $M, w \models \varphi$?

Algorithm: LTL-$MC(\varphi, M, w)$ – *game semantics*

case
φ propositional: return $\pi(w) \models \varphi$
$\varphi = \theta_1 \vee \theta_2$: ($\exists$-branch) return LTL-$MC(\theta_i, M, w)$
$\varphi = \theta_1 \wedge \theta_2$: ($\forall$-branch) return LTL-$MC(\theta_i, M, w)$
$\varphi = next\ \psi$: return LTL-$MC(\psi, M, R(w))$
$\varphi = \theta\ until\ \psi$: return LTL-$MC(\psi, M, w)$ or return
(LTL-$MC(\theta, M, w)$ and LTL-$MC(\theta\ until\ \psi, M, R(w))$
)
esac.

But: When does the game end?

From Finite to Infinite Games

Problem: Algorithm may not terminate!!!

Solution: Redefine games

- Standard alternation is a *finite* game between $\exists$ and $\forall$.

- Here we need an *infinite* game.

- In an infinite play $\exists$ needs to visit non-$until$ formulas infinitely often – “not get stuck in one $until$ formula”.

Büchi Alternation Muller&Schupp, 1985:

- Infinite computations allowed

- On infinite computations $\exists$ needs to visit accepting states ∞ often.

Lemma: Büchi-ALOGSPACE=PTIME

Corollary: LTL-MC $\in$ PTIME

LTL Satisfiability

Hope: Use Büchi alternation to adapt K-SAT to LTL-SAT.

Problems:

- What is time bounded Büchi alternation? Büchi-PTIME?

- Successors cannot be split!

$$\begin{tikzcd} & next\ \delta, next\ \eta \arrow[dl] \arrow[dr] & \\ \delta & & \eta \end{tikzcd}$$

Alternating Automata

Alternating automata: 2-player games

Nondeterministic transition: $\rho(s,a) = t_1 \vee t_2 \vee t_3$

Alternating transition: $\rho(s,a) = (t_1 \wedge t_2) \vee t_3$
"either both t_1 and t_2 accept or t_3 accepts".

- $(s,a) \mapsto \{t_1, t_2\}$ or $(s,a) \mapsto \{t_3\}$
- $\{t_1, t_2\} \models \rho(s,a)$ and $\{t_3\} \models \rho(s,a)$

Alternating transition function: $\rho : S \times \Sigma \rightarrow \mathcal{B}^+(S)$
(positive Boolean formulas over S)

- $P \models \rho(s,a)$ – P *satisfies* $\rho(s,a)$
  - $P \models$ **true**
  - $P \not\models$ **false**
  - $P \models (\theta \vee \psi)$ if $P \models \theta$ or $P \models \psi$
  - $P \models (\theta \wedge \psi)$ if $P \models \theta$ and $P \models \psi$

Alternating Automata on Finite Words

Brzozowski&Leiss, 1980: Boolean automata

$A = (\Sigma, S, s_0, \rho, F)$

- Σ, S, $F \subseteq S$: as before
- $s_0 \in S$: initial state
- $\rho : S \times \Sigma \to \mathcal{B}^+(S)$: alternating transition function

Game:

- Board: $a_0, \ldots, a_{n-1}$
- Positions: $S \times \{0, \ldots, n-1\}$
- Initial position: $(s_0, 0)$
- Automaton move at (s, i):
choose $T \subseteq S$ such that $T \models \rho(s, a_i)$
- Opponent's response:
move to $(t, i+1)$ for some $t \in T$
- Automaton wins at (s', n) if $s' \in F$

Acceptance: Automaton has a winning strategy.

Expressiveness

Expressiveness: ability to recognize sets of "boards", i.e., languages.

BL'80,CKS'81:

- Nondeterministic automata: regular languages
- Alternating automata: regular languages

What is the point?: Succinctness

Exponential gap:

- Exponential translation from alternating automata to nondeterministic automata
- In the worst case this is the best possible

Crux: 2-player games $\mapsto$ 1-player games

Alternating Büchi Automata

$A = (\Sigma, S, s_0, \rho, F)$

Game:

- *Infinite* board: $a_0, a_1 \ldots$
- Positions: $S \times \{0, 1, \ldots\}$
- Initial position: $(s_0, 0)$
- Automaton move at (s, i):
choose $T \subseteq S$ such that $T \models \rho(s, a_i)$
- Opponent's response:
move to $(t, i+1)$ for some $t \in T$
- Automaton wins if play goes through infinitely many positions (s', i) with $s' \in F$

Acceptance: Automaton has a winning strategy.

Example

$A = (\{0, 1\}, \{m, s\}, m, \rho, \{m\})$

- $\rho(m, 1) = m$
- $\rho(m, 0) = m \wedge s$
- $\rho(s, 1) = \mathbf{true}$
- $\rho(s, 0) = s$

Intuition:

- m is a master process. It launches s when it sees 0.
- s is a slave process. It wait for 1, and then terminates successfully.

$L(A) =$ infinitely many 1's.

Expressiveness

Miyano&Hayashi, 1984:

- Nondeterministic Büchi automata: ω-regular languages
- Alternating automata: ω-regular languages

What is the point?: Succinctness

Exponential gap:

- Exponential translation from alternating Büchi automata to nondeterministic Büchi automata
- In the worst case this is the best possible

Back to LTL

Old temporal structure: $M = (W, R, \pi)$

- W: worlds
- $R : W \rightarrow W$: successor function
- $\pi : W \rightarrow 2^{Prop}$: truth assignments

New temporal structure: $\sigma \in (2^{Prop})^\omega$ (unwind the function R)

Temporal Semantics: $models(\varphi) \subseteq (2^{Prop})^\omega$

Theorem[V., 1994] : For each LTL formula φ there is an alternating Büchi automaton A_φ with $||\varphi||$ states such that $models(\varphi) = L(A_\varphi)$.

Intuition: Consider LTL-MC as an alternating Büchi automaton.

Alternating Automata Nonemptiness

Given: Alternating Büchi automaton A

Two-step algorithm:

- Construct *nondeterministic Büchi automaton* A^n such that $L(A^n) = L(A)$ (exponential blow-up)
- Test $L(A^n) \neq \emptyset$ (NLOGSPACE)

Problem: A^n is exponentially large.

Solution: Construct A^n *on-the-fly*.

Corollary 1: Alternating Büchi automata nonemptiness is in PSPACE.

Corollary 2: LTL satisfiability is in PSPACE [Halpern&Reif, 1982, Sistla&Clarke, 1982].

Back to Trees

Games, via alternating automata, provide the key to obtaining elementary decision procedures to numerous, modal, temporal, and dynamic logics.

Theorem [Kupferman&V.&Wolper, 1994]: For each CTL formula φ there is an alternating Büchi tree automaton A_φ with $||\varphi||$ states such that $models(\varphi) = L(A_\varphi)$.

Theorem [KVW, 1986]: There is an exponential translation of alternating Büchi tree automata to nondeterministic Büchi tree automata.

Proposition: Nonemptiness of nondeterministic Büchi tree automata can be checked in quadratic time [V.&Wolper, 1984]

Corollary: There is an exponential algorithm for satisfiability of CTL [Emerson&Halpern, 1985]

Satisfiability of the μ-calculus

Fixpoints:

- *Least fixpoint*: finite recurrence
- *Greatest fixpoint*: infinite recurrence
- *Nested fixpoints*: alternation of finite and infinite recurrence!

Parity Acceptance Condition

- $\mathcal{F} = (F_1, F_2, \ldots, F_k)$ - partition of state set S.
- *Parity index*: k
- *Acceptance*: Least i such that F_i is visited infinitely often is *even*.

Theorem [Kupferman&V.&Wolper, 1994]:
For each μ-calculus formula φ there is an alternating parity tree automaton A_φ with $||\varphi||$ states and index $||\varphi||$ such that $models(\varphi) = L(A_\varphi)$.

Application to μ-Calculus

Theorem [Müller&Schup, 1995]: There is an exponential translation of alternating parity tree automata to nondeterministic parity tree automata.
Corollary: There is an exponential algorithm for satisfiability of the μ-calculus [Emerson&Jutla, 1989]

Apologetic Note: Glossed over nonemptiness problem for nondeterministic parity tree automata – "kind of polynmial time"

μ-Calculus with Inverse Roles

Up and Down the Tree:

- $\langle R \rangle$ – go *down* the tree
- $\langle R^- \rangle$ – go *up* the tree

Automata-Theoretic Analog: two-way tree automata, which go up and down the tree!

- Extend two-way word automata from [Rabin&Scott, 1959]

Theorem [V., 1998]: There is an exponential translation of two-way alternating parity tree automata to one-way nondeterministic parity tree automata.
Corollary: There is an exponential algorithm for satisfiability of the μ-calculus with inverse roles [V., 1998]

Dealing with Nominals and Graded Modalities

Basic Idea: more powerful automata!

- Dealing with graded modalities: graded automaton transitions – “accept from state s a tree node labeled with a if at least five child node are accepted from state t”
- Dealing with nominals: from trees to forests – let nominals serve as roots of trees

Decidability results: EXPTIME

- Handling both inverse roles and nominals [Sattler&V., 2001]
- Handling both inverse roles and graded modalities [Bonatti, Lutz, Murano &V., 2006]
- Handling both nominals and graded modalities [Bonatti, Lutz, Murano &V., 2006]

A Recipe for Decision Procedures

A Simple Recipe

- Prove tree-model property
  - *Note*: fails for μ-calculus with inverse roles, graded modalities, and nominals!
- Define the proper variant of alternating parity tree automata.
- Prove linear translation from logic to automata.
- Prove exponential translation to nondeterministic parity tree automata.

Discussion

Major Points:

- The *logic-automata connection* is one of the most fundamental paradigms of logic.
- One of the major benefits of this paradigm is its algorithmic consequences.
- A newer component of this approach is that of *games*, and *alternating automata* as their automata-theoretic counterpart.
- The interaction between logic, automata, games. and algorithms yields a fertile research area.

Tower of Abstractions

Key idea in science: *abstraction tower*

strings

quarks

hadrons

atoms

molecules

amino acids

genes

genomes

organisms

populations

Abstraction Tower in CS

CS Abstraction Tower:

analog devices

digital devices

microprocessors

assembly languages

high-level language

libraries

software frameworks

Crux: Abstraction tower is the only way to deal with complexity!

Similarly: We need high-level algorithmic building blocks, e.g., *BFS*, *DFS*.

This talk: *Games/alternation* as a high-level algorithmic construct.

Alternation

Two perspectives:

- Two-player games
- Control mechanism for parallel processing

Two Applications:

- Model checking
- Satisfiability checking

Bottom line: Alternation is a key algorithmic construct in automated reasoning — used in industrial tools.

Question: Is it not time to implement decision procedures for expressive hybrid logic?